

AD-A038 892

NAVAL SEA SYSTEMS COMMAND WASHINGTON D C

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAR - IN SUP--ETC(U)

F/G 17/1

DEC 75

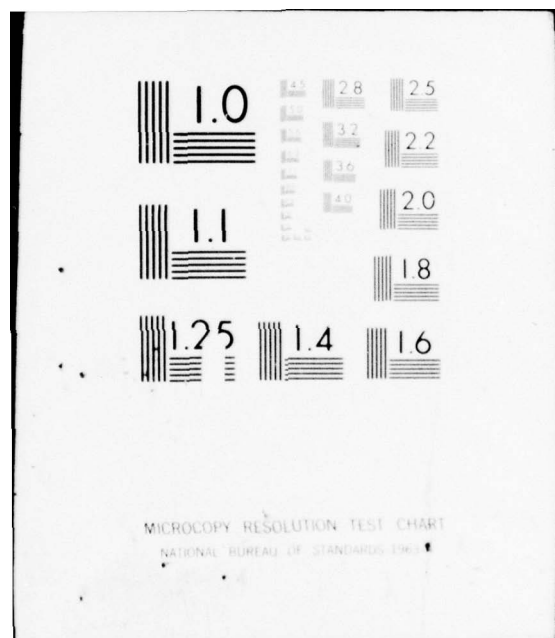
UNCLASSIFIED

NAVSEA-06H1/036-EVA/MOST-

NL

1 OF 2
AD
A038892





SEA 06H1/036-EVA/MOST-5

1 DECEMBER 1975

**NAVSEA OCEAN ENVIRONMENTAL
ACOUSTIC DATA BANK**

—NAVDAB—

**IN SUPPORT OF
MOBILE SONAR TECHNOLOGY DEVELOPMENT**

VOLUME 4



**NAVAL SEA SYSTEMS COMMAND
DEPARTMENT OF THE NAVY
WASHINGTON, D.C. 20362**

Approved for public release; distribution unlimited.

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

014
ADA 038892

AD No. _____
DDC FILE COPY

DDC
RECEIVED
APR 28 1977
B

1
SEA 06H1/036-EVA/MOST-5

1 DECEMBER 1975

**NAVSEA OCEAN ENVIRONMENTAL
ACOUSTIC DATA BANK
—NAVDAB—**

**IN SUPPORT OF
MOBILE SONAR TECHNOLOGY DEVELOPMENT**

VOLUME 4 ✓



**NAVAL SEA SYSTEMS COMMAND
DEPARTMENT OF THE NAVY
WASHINGTON, D.C. 20362**

SUBMISSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY CODES	
DISC	AVAIL. 20362 SPECIAL
A	

DDC
RECEIVED
APR 28 1977
A

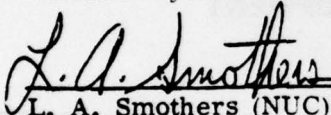
Approved for public release; distribution unlimited.

ADMINISTRATIVE STATEMENT

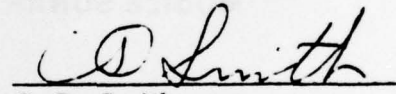
The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is a joint effort of the Naval Undersea Center (NUC), the Naval Underwater Systems Center (NUSC), the Naval Research Laboratory (NRL), and the Naval Oceanographic Office (NAVOCEANO), under the sponsorship of the Sonar Technology Office of the Naval Sea Systems Command (Task Area SF 52 552 601). Each of these organizations is represented in the NAVDAB Steering Group. The Steering Group has the responsibility for development of the data bank, which has been installed at NUC/San Diego, California, NUSC/New London, Connecticut, and NAVOCEANO/Washington, D.C.

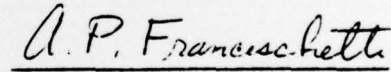
The NAVDAB Steering Group is grateful to G. F. Anderson of Computer Sciences Corporation and G. E. Miller of Arthur D. Little, Inc., for their assistance on this project.

Released by:


L. A. Smothers (NUC)
NAVDAB Chairman

Sponsoring Authority:

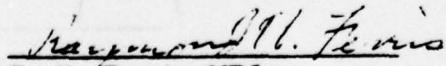

C. D. Smith
NAVSEA

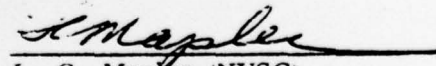

A. P. Franceschetti
NAVSEA

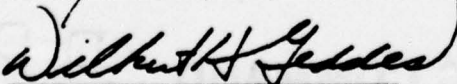
Steering Group

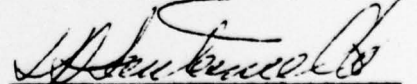

F. R. DiNapoli (NUSC)

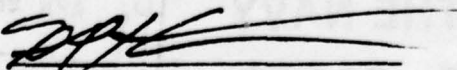

K. V. Mackenzie (NAVOCEANO)

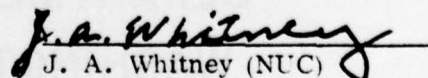

R. H. Ferris (NRL)

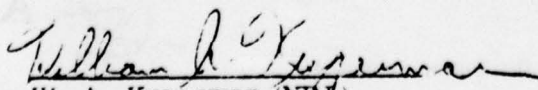

L. C. Maples (NUSC)


W. H. Geldes (NAVOCEANO)


S. R. Santaniello (NUSC)


R. F. Hosmer (NUC)


J. A. Whitney (NUC)


W. A. Kuperman (NRL)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NAVSEA 06H1/036-EVA/MOST-5	2. GOVT ACCESSION NO.	3. RECIPIENT CATALOG NUMBER (9)
4. TITLE (and Subtitle) NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUPPORT OF MOBILE SONAR TECHNOLOGY DEVELOPMENT, VOLUME 4,	5. TYPE OF REPORT & PERIOD COVERED Final Rept.	
6. AUTHOR(s)	7. PERFORMING ORG. REPORT NUMBER	
NAVDAB Steering Group	SF 52-552-601	
8. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Undersea Center, Code 3073 San Diego, CA. 92132	9. PROGRAM ELEMENT PROJECT, TASK AREA & WORK UNIT NUMBERS Task No. 19324	
10. CONTROLLING OFFICE NAME AND ADDRESS Naval Sea Systems Command (NAVSEA 06H1-4) Washington, D.C. 20362	11. REPORT DATE 10 1 December 1975	
12. MONITORING AGENCY NAME & ADDRESS (if not from Controlling Office) (12) 162P.	13. SECURITY CLASS. (of this report) Unclassified	
14. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
15. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) (16) F52552 (17) SF52552601		
16. SUPPLEMENTARY NOTES Prepared in cooperation with the NAVDAB Steering Group.		
17. KEY WORDS (Continue on reverse side if none; use block number) Data Bank Underwater Acoustics Environmental data Computer Programs		
18. ABSTRACT (Continue on reverse side if none; use block number) The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base of underwater acoustic and associated environmental data. This document provides detailed information on the retrieval phase computer programs.		

DD FORM 1473 EDITION OF 1 NOV 55 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391345

Y/B

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

1. TITLE	
2. AUTHOR	
3. PERIODICITY	
4. NUMBER OF PAGES	
5. DATE	
6. ABSTRACT	
7. SUMMARY	
8. CONCLUSIONS	
9. REFERENCES	
10. NOTES	
11. COMMENTS	
12. INDEXING	
13. DISTRIBUTION	
14. SECURITY CLASSIFICATION	
15. OTHER INFORMATION	

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)



DEPARTMENT OF THE NAVY
NAVAL SEA SYSTEMS COMMAND
WASHINGTON, D.C. 20362

FOREWORD

The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar application. NAVDAB is unique in that it is primarily for environmental and acoustic data taken concurrently and it is the only data bank of this type operated for mobile sonar application.

A set of five reports have been prepared to describe how NAVDAB works and how to use it. This is volume 4 of the set. The individual reports cover the following topics:

- Volume 1 User's Guide to Retrieval
- Volume 2 Input Format Guide
- Volume 3 Details of Creation Phase
- Volume 4 Details of Retrieval Phase
- Volume 5 Details of Miscellaneous Support Programs

Governmental and industrial activities desiring to submit or retrieve environmental acoustic data should contact:

Naval Undersea Center
Code 3073
San Diego, CA - 92132

Naval Underwater Systems Center
Code TALA
New London Laboratory
New London, CT - 06320

Naval Oceanographic Office
Code 3440
Washington, D.C. - 20373

Copies of the five reports can be obtained through the Defense Documentation Center, Defense Supply Agency, Cameron Station, Alexandria, VA - 22314.

C.D. Smith, Director
Sonar Technology Office,
06H1/036

CONTENTS

<u>Section</u>		<u>Page</u>
1.0	INTRODUCTION	1-1
2.0	DESCRIPTION OF THE DATA BASE	2-1
3.0	DESCRIPTION OF THE RETRIEVAL PROCESS	3-1

Appendices

A.	Computer Program Listings	A-1
B	Current NAVDAB Steering Group	B-1

LIST OF ILLUSTRATIONS

<u>Figure</u>		
2-1	NAVDAB Data Base Structure	2-2
3-1 } 3-2 } 3-3 }	Flowcharts for Retrieval Phase	{ 3-2 3-3 3-4
A-1	Retrieval Program Cross-Reference Map	A-4

SECTION 1.0

INTRODUCTION

The Naval Sea Systems Command (NAVSEA) Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar applications. The purpose of this document is to provide details of the computer programs which access the NAVDAB data base and retrieve data from it.

This is Volume 4 of the five-volume set covering the NAVDAB computer program documentation. The individual volumes cover the following topics:

Volume 1. User's Guide to Retrieval

Volume 2. Input Format Guide

Volume 3. Details of Creation Phase

Volume 4. Details of Retrieval Phase

Volume 5. Details of Miscellaneous Support Programs

Copies of these documents may be obtained from the Defense Documentation Center.

SECTION 2.0

DESCRIPTION OF THE DATA BASE

Many factors affect the design of a data base and retrieval system. In the case of NAVDAB a large number and variety of data parameters were considered. To meet this requirement an open-ended tabular format, with variable record length, was selected to accommodate the initial sets of parameters and data and allow for expansion.

Another primary consideration in the data base design was the need for rapid access to all elements of all data. The data base must be suited to a versatile, efficient, random-access, retrieval system. Random-access retrieval techniques dictate, to a great extent, how data are structured on a storage device.

Other requirements of the NAVDAB system include: safeguard of classified information; allowance for operation in either batch or time-sharing mode; ease of updating and maintenance; and use of a universally-accepted computer language, as well as sufficient modularity, for ready adaptation to the different types of computers on which the system may be implemented.

In view of all these considerations, the data organization shown in Figure 2-1 was developed. It is a multilevel, hierarchically-organized, partially circular, linked-list structure, with large-capacity disk for primary storage. Four levels are involved:

1. **EXPERIMENT** — The highest level, pertains to the overall program of measurements — such as AMOS, FASOR I (Forward Area Sonar Research)
2. **CRUISE** — The second level is applicable to natural subsets of an **EXPERIMENT**, such as individual AMOS cruises or measurements taken within a specified area

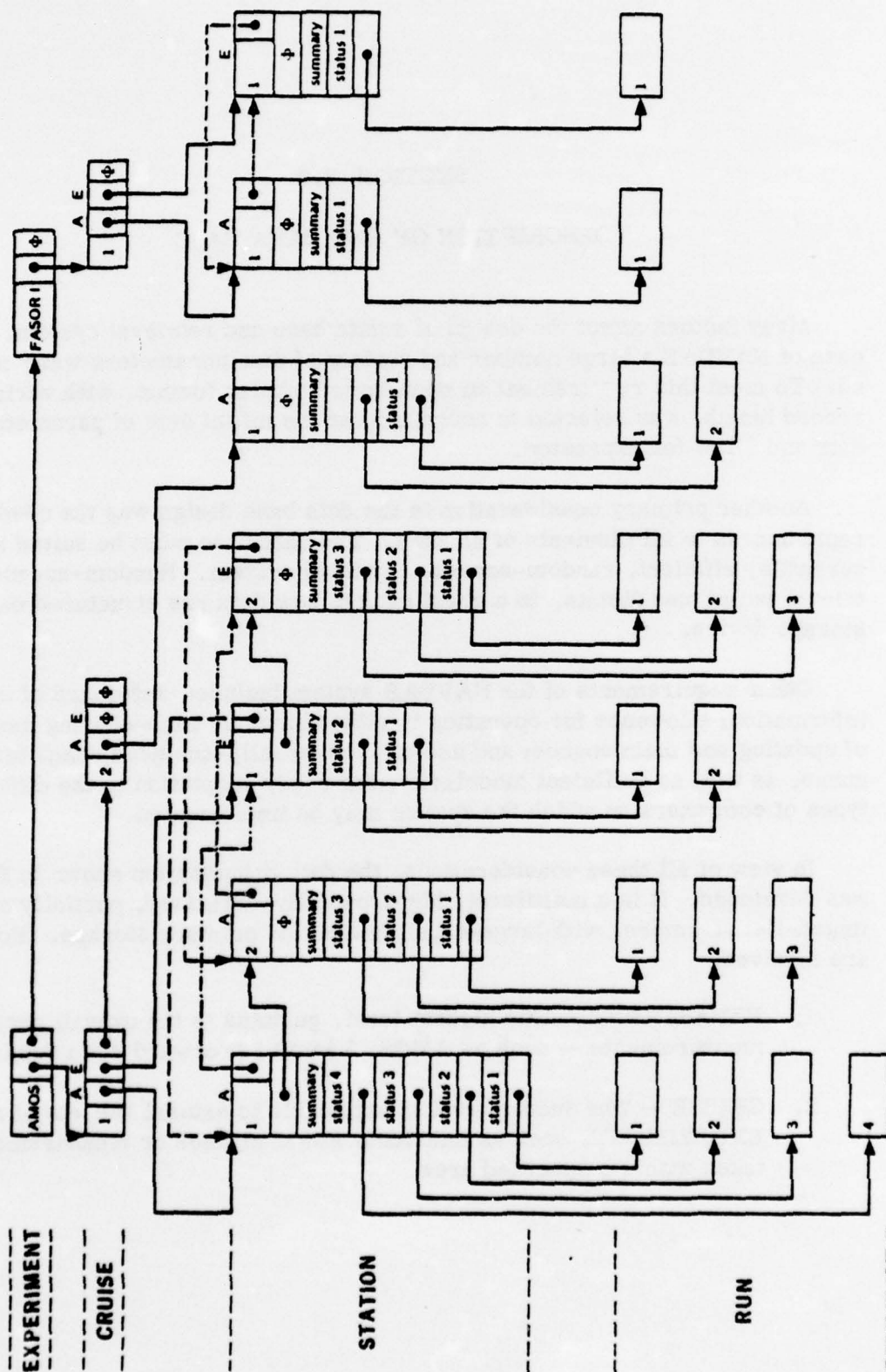


Figure 2-1. NAVDAB Data Base Structure.

3. STATION - A subdivision of a CRUISE: refers to measurements at or about a specific geographical location, such as FASOR II, Cruise 1, Station 1
4. RUN - The fourth and lowest level: designates a discrete set of measurements forming an element of a STATION, such as a propagation loss run or a hydrographic cast

Actual numerical data are stored only at the RUN level, which contains no other information. Key catalog information about the various Runs of a Station are stored at the STATION level, which contains, for all the Runs, chronological and geographical data and access category, and lists the acoustic and environmental parameters. The organization of the data is described for each Run, including the Run's storage location on the disk. Locations of associated Stations are also included. There are two types of Stations, acoustic and environmental, but the only environmental data currently approved by the Steering Group are those supporting accepted acoustic data.

Catalog information for the Stations of a Cruise is stored at the CRUISE level, which also contains storage locations of the first acoustic and environmental Stations and the next Cruise of the Experiment. In addition, there is an alphanumeric description of the features of the particular Cruise. These notes may be printed out in the retrieval process as background information.

The EXPERIMENT level contains summary information for the Experiment and directs flow down through the other levels. This flexible pointer structure allows additions or deletions of data without changing the programs that access the data.

SECTION 3.0

DESCRIPTION OF THE RETRIEVAL PROCESS

The retrieval process consists of two operations, a data selection pass and an application pass. The two-pass technique was selected to allow flexibility in the type of request for data in terms of area, season, frequency, water depth, range, propagation mode, etc.

The data selection pass reads and analyzes the request and, starting at the EXPERIMENT level, examines the summary data stored at each level to determine whether there is justification to continue searching at the lower levels or to go "across" to the next entry at the same level. The examination continues to the STATION level, where the final decision is made as to whether any of the Runs of the Station meet the acoustic data requirements of the potential user.

When a suitable acoustic Run is found, the pointer to the associated environmental Station is followed to determine whether environmental requirements have been met. If both acoustic and environmental criteria have been met, the pointers to data locations are stored in a temporary file for processing by the application pass which reads in the pointers and retrieves the data. A standard set of application programs is planned to output the data in a format selected by the user. The user may also receive the output with conversion to units of his choice.

Flowcharts for the retrieval process are shown in Figure 3-1 through 3-3. Complete computer program listings are contained in Appendix A.

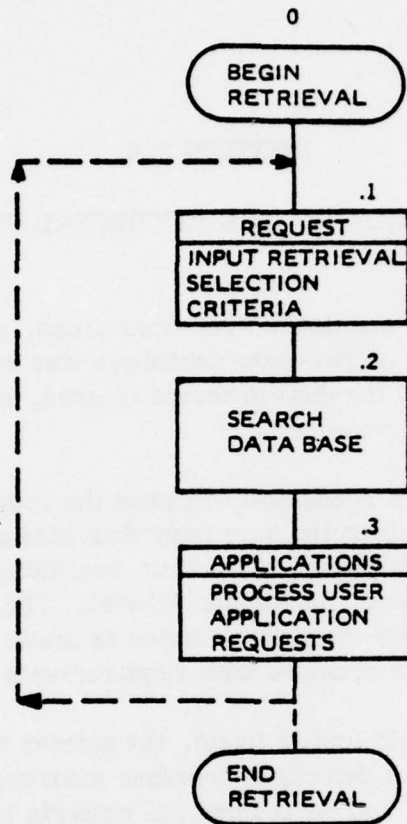


Figure 3-1.

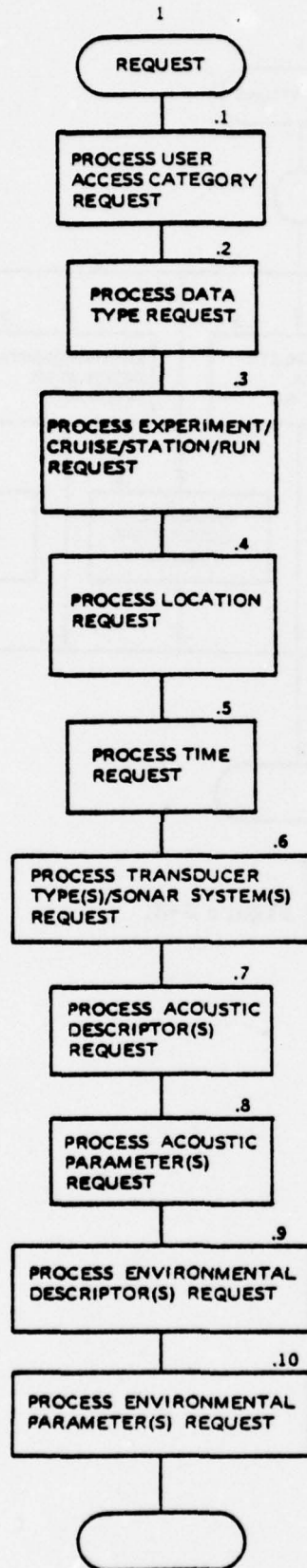


Figure 3-2.

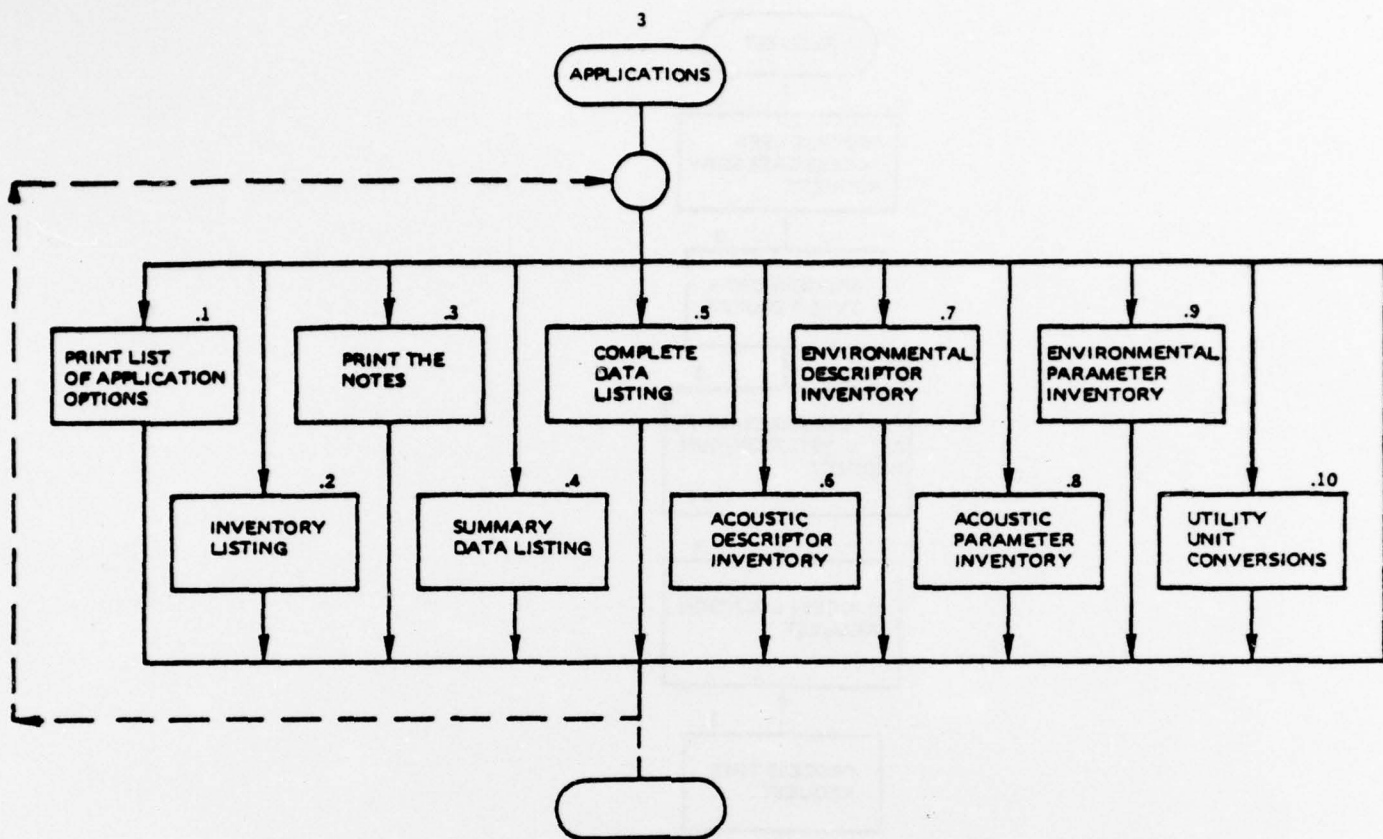


Figure 3-3.

APPENDIX A
COMPUTER PROGRAM LISTINGS

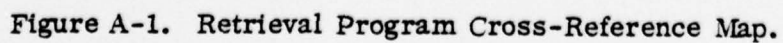
ELEMENT TABLE

NAME	PAGE	TYPE	DATE	TIME	SEQ #	SIZE-PR,TEXT	(CYCLE WORD)	PSRMODE	LOCATION
ACPRCS	A-5	FOR SYMB	22 AUG 74	10:23:45	1	12	5	0	1792
ACPRCS		RELOCATABLE	16 DEC 75	16:07:104	2	1	8		1804
ANGLES	A-7	FOR SYMB	01 NOV 74	12:02:12	3	13	5	0	1813
ANGLES		RELOCATABLE	16 DEC 75	16:07:101	4	1	10		1826
AREACK	A-9	ELT SYMB	07 NOV 75	15:41:24	5	3	5	0	1837
AREACK		RELOCATABLE	16 DEC 75	16:07:112	6	1	5		1840
ASUNPK	A-10	FOR SYMB	16 DEC 75	10:16:40	7	7	5	0	1846
ASUNPK		RELOCATABLE	16 DEC 75	16:07:113	8	1	4		1853
AUNPK	A-11	FOR SYMB	16 DEC 75	10:17:48	9	32	5	0	1858
AUNPK		RELOCATABLE	16 DEC 75	16:07:124	10	2	17		1890
CHECK	A-14	FOR SYMB	10 SEP 75	11:51:04	11	9	5	0	1909
CHECK		RELOCATABLE	16 DEC 75	16:07:37	12	1	4		1918
CLASS	A-15	FOR SYMB	16 DEC 75	09:48:12	13	54	5	0	1923
CLASS		RELOCATABLE	16 DEC 75	16:07:49	14	2	54		1981
CONVNT	A-21	FOR SYMB	31 OCT 75	15:40:50	15	27	5	0	2037
CONVNT		RELOCATABLE	16 DEC 75	16:08:01	16	3	25		2064
CVTPRO	A-24	FOR SYMB	21 OCT 74	10:18:36	17	38	5	0	2092
CVTPRO		RELOCATABLE	16 DEC 75	16:08:04	18	3	28		2130
CYCLES	A-28	FOR SYMB	24 JUN 74	14:33:34	19	12	5	0	2161
CYCLES		RELOCATABLE	16 DEC 75	16:08:01	20	1	8		2173
DATECK	A-30	FOR SYMB	16 DEC 75	10:24:48	21	10	5	0	2182
DATECK		RELOCATABLE	16 DEC 75	16:08:05	22	1	12		2192
DATLST	A-31	FOR SYMB	04 NOV 75	17:21:46	23	83	5	0	2205
DATLST		RELOCATABLE	16 DEC 75	16:08:13	24	3	79		2288
DBLOSS	A-38	FOR SYMB	06 SEP 74	09:20:35	25	15	5	0	2370
DBLOSS		RELOCATABLE	16 DEC 75	16:08:17	26	1	9		2385
DEBEE	A-40	FOR SYMB	30 OCT 75	08:46:39	27	19	5	0	2395
DEBEE		RELOCATABLE	16 DEC 75	16:08:28	28	1	15		2414
DENSTY	A-42	FOR SYMB	28 JUN 74	13:34:17	29	12	5	0	2430
DENSTY		RELOCATABLE	16 DEC 75	16:08:25	30	1	7		2442
DESCIB	A-44	FOR SYMB	02 OCT 75	09:10:28	31	5	5	0	2450
DESCIB		RELOCATABLE	16 DEC 75	16:08:28	32	1	5		2455
DSKRED	A-45	FOR SYMB	16 DEC 75	10:24:49	33	3	5	0	2461
DSKRED		RELOCATABLE	16 DEC 75	16:08:37	34	1	4		2464
EAST	A-46	ELT SYMB	04 NOV 75	17:05:22	35	2	5	0	2469
EAST		RELOCATABLE	16 DEC 75	16:08:36	36	1	3		2471
ENERGY	A-47	FOR SYMB	24 JUN 74	14:33:40	37	13	5	0	2475
ENERGY		RELOCATABLE	16 DEC 75	16:08:40	38	1	8		2488
FREEFO	A-49	FOR SYMB	16 DEC 75	10:30:34	39	17	5	0	2497
FREEFO		RELOCATABLE	16 DEC 75	16:08:48	40	1	16		2514
GETINT	A-51	FOR SYMB	04 NOV 75	17:26:24	41	30	5	0	2531
GETINT		RELOCATABLE	16 DEC 75	16:09:34	42	3	16		2561
IDNAME	A-55	SYMBOLIC	16 OCT 75	10:43:27	43	10	5	0	2580
IDNAME		RELOCATABLE	16 DEC 75	16:09:05	44	1	10		2590
INPUT	A-56	FOR SYMB	16 DEC 75	10:26:04	45	180	5	0	2601
INPUT		RELOCATABLE	16 DEC 75	16:09:28	46	2	160		2781
INTENS	A-73	FOR SYMB	24 JUN 74	14:34:29	47	12	5	0	2943
INTENS		RELOCATABLE	16 DEC 75	16:09:28	48	1	7		2955
INHTY	A-75	FOR SYMB	14 AUG 75	10:17:53	49	12	5	0	2963
INHTY		RELOCATABLE	16 DEC 75	16:09:40	50	2	10		2975
LINEAR	A-76	FOR SYMB	30 AUG 74	13:34:51	51	15	5	0	2987
LINEAR		RELOCATABLE	16 DEC 75	16:09:41	52	1	11		3002
LNTHNE	A-78	FOR SYMB	24 JUN 74	14:32:36	53	12	5	0	3014
LNTHNE		RELOCATABLE	16 DEC 75	16:09:53	54	1	8		3026

PAGE

MSQVT	A-80	ELT SYMB	07 NOV 75	15:41:25	55		12	5	0	1	3035
MSQVT		RELOCATABLE	16 DEC 75	16:10:01	56	1	7				3047
NAMEIT	A-82	FOR SYMB	03 SEP 75	11:54:31	57		49	5	0	1	3055
NAMEIT		RELOCATABLE	16 DEC 75	16:10:05	58	1	27				3124
NAMEIT	A-87	FOR SYMB	02 SEP 75	15:11:05	59		13	5	0	1	3152
NAMEIT		RELOCATABLE	16 DEC 75	16:10:04	60	1	15				3165
NOTE	A-89	FOR SYMB	16 DEC 75	10:27:01	61		17	5	0	1	3181
NOTE		RELOCATABLE	16 DEC 75	16:10:29	62	2	14				3198
PARATB	A-90	FOR SYMB	02 OCT 75	10:15:31	63		7	5	0	1	3214
PARATB		RELOCATABLE	16 DEC 75	16:10:36	64	1	4				3221
PDP	A-91	FOR PROC	24 OCT 75	12:08:58	65		14	1	0	1	3228
PERCNT	A-92	FOR SYMB	17 JUL 75	13:52:12	66		12	5	0	1	3242
PERCNT		RELOCATABLE	16 DEC 75	16:10:41	67	1	8				3254
PLANAR	A-94	FOR SYMB	03 SEP 75	09:15:53	68		13	5	0	1	3263
PLANAR		RELOCATABLE	16 DEC 75	16:10:48	69	1	9				3276
PO-0AR	A-96	FOR SYMB	24 JUN 74	14:34:03	70		12	5	0	1	3286
PO-0AR		RELOCATABLE	16 DEC 75	16:10:51	71	1	7				3298
PRESSR	A-98	FOR SYMB	22 AUG 74	10:25:08	72		14	5	0	1	3306
PRESSR		RELOCATABLE	16 DEC 75	16:11:03	73	1	10				3320
PRSLVL	A-101	FOR SYMB	22 AUG 74	11:09:28	74		12	5	0	1	3331
PRSLVL		RELOCATABLE	16 DEC 75	16:11:01	75	1	7				3343
RANGE	A-103	FOR SYMB	16 DEC 75	10:27:24	76		37	5	0	1	3351
RANGE		RELOCATABLE	16 DEC 75	16:11:04	77	2	23				3388
SELECT	A-106	FOR SYMB	16 DEC 75	10:27:49	78		139	5	0	1	3413
SELECT		RELOCATABLE	16 DEC 75	16:11:15	79	5	92				3552
SHORT	A-110	FOR SYMB	21 AUG 75	15:07:12	80		11	5	0	1	3649
SHORT		RELOCATABLE	16 DEC 75	16:11:26	81	2	9				3660
SHRTME	A-121	FOR SYMB	24 JUN 74	14:32:29	82		12	5	0	1	3671
SHRTME		RELOCATABLE	16 DEC 75	16:11:24	83	1	9				3683
SORGET	A-123	FOR SYMB	22 AUG 74	11:07:41	84		16	5	0	1	3693
SORGET		RELOCATABLE	16 DEC 75	16:11:29	85	1	9				3709
SPEEDS	A-125	FOR SYMB	26 JUN 74	21:22:11	86		13	5	0	1	3719
SPEEDS		RELOCATABLE	16 DEC 75	16:11:43	87	1	10				3732
STACK	A-127	ELT SYMB	04 NOV 75	17:05:13	88		3	5	0	1	3743
STACK		RELOCATABLE	16 DEC 75	16:11:40	89	1	3				3746
STORE	A-128	SYMBOLIC	10 OCT 75	15:17:14	90		17	5	0	1	3750
STORE		RELOCATABLE	16 DEC 75	16:11:51	91	2	4				3758
THPTUR	A-129	FOR SYMB	17 JUL 75	13:45:33	92		13	5	0	1	3766
THPTUR		RELOCATABLE	16 DEC 75	16:12:01	93	1	11				3779
UNPKCR	A-131	ELT SYMB	04 NOV 75	17:05:15	94		10	5	0	1	3791
UNPKCR		RELOCATABLE	16 DEC 75	16:12:03	95	1	4				3801
UNPKXP	A-132	ELT SYMB	07 NOV 75	15:41:20	96		7	5	0	1	3808
UNPKXP		RELOCATABLE	16 DEC 75	16:12:12	97	1	5				3815
UNTCVT	A-133	FOR SYMB	31 OCT 75	17:04:53	98		11	5	0	1	3821
UNTCVT		RELOCATABLE	16 DEC 75	16:12:25	99	2	8				3832
UPKREA	A-134	ELT SYMB	07 NOV 75	15:41:22	100		15	5	0	1	3842
UPKREA		RELOCATABLE	16 DEC 75	16:12:28	101	1	8				3857
VOLUME	A-137	FOR SYMB	24 JUN 74	14:32:21	102		13	5	0	1	3866
VOLUME		RELOCATABLE	16 DEC 75	16:12:39	103	1	8				3879
WEST	A-139	ELT SYMB	04 NOV 75	17:06:04	104		2	5	0	1	3888
WEST		RELOCATABLE	16 DEC 75	16:12:38	105	1	3				3890
											3894

CALLING ROUTINES



BFOR,SA N,ACPRES,,ACPRES

```

1. SUBROUTINE ACRES(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF ACRES MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(J,J),UNITS(I),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(J)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(I,I),I=1,3)/49,99,50/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(I,J),I=2,3),J=1,3)/'MICRO ','BARS ',
18. C 'PASCAL','S',
19. C ENTER INTERNAL UNITS.
20. C
21. C UNITS(3)='MICRO '
22. C UNITS(4)='PASCAL'
23. C
24. C IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
25. C
26. C IF UNICDE.EQ.0) UNICDE=-50
27. C
28. C SET CONVERSION FACTORS.
29. C
30. C DATA (FACTOR(I),I=1,3)/ 1.05,1.03,100/
31. C
32. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
33. C CODE INTO 'UNITS'.
34. C
35. C ICODE=ABS(UNICDE)
36. C DO 2 I=1,3
37. C IF ICODE.NE.ENTRY(I,I) GO TO 2
38. C IF UNICDE.EQ.0) VALUE=VALUE*FACTOR(I)
39. C IF UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
40. C DO 1 J=1,2
41. C UNITS(J)=ENTRY(J,I,I)
42. C 1 CONTINUE
43. C
44. C AT THIS POINT, CONVERSION IS COMPLETE.
45. C
46. C RETURN
47. C
48. C ELSE, CHECK REST OF TABLE.
49. C
50. C 2 CONTINUE
51. C
52. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
53. C
54. C

```

```

55. UNITS(1)=UNITS *
56. UNITS(2)=ERROR *
57. PRINT 101,ICODE
58. 101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ACOUSTIC PRESSURE TABLE.')
```

```

59. RETURN
60. END
```

9FOR,50 H,ANGLES,ANGLES

```

1. SUBROUTINE ANGLES(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF ANGLES MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(3)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,3)/37,38,85/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,1),1=2,3),1=1,3)/.RADIANS,15
18. C .DEGREE,15 .FTMMS/1,INTCLML/
19. C
20. C SET CONVERSION FACTORS.
21. C
22. C DATA (FACTOR(1,1=1,2)/1.000,.017453291700/
23. C
24. C ENTER INTERNAL DEGREE UNITS.
25. C
26. C UNITS(3)=.DEGREE
27. C UNITS(4)=15
28. C
29. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
30. C
31. C IF (UNICODE.EQ.0) UNICODE=-37
32. C
33. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
34. C CODE INTO UNITS.
35. C
36. C ICODE=ABS(UNICODE)
37. C DO 2 I=1,2
38. C IF (ICODE.NE.ENTRY(1,1)) GO TO 2
39. C IF (UNICODE.GE.0) VALUE=VALUE*FACTOR(1)
40. C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(1)
41. C DO 1 J=1,2
42. C UNITS(J)=ENTRY(J+1,1)
43. C 1 CONTINUE
44. C
45. C AT THIS POINT, CONVERSION IS COMPLETE.
46. C
47. C RETURN
48. C
49. C ELSE, CHECK REST OF TABLE.
50. C
51. C 2 CONTINUE
52. C IF (ICODE.NE.ENTRY(1,1)) GO TO 3
53. C IF (UNICODE.GE.0) VALUE=ATAN(VALUE/1000.1)/.0174533
54. C IF (UNICODE.LT.0) VALUE=ATAN(VALUE*.0174533)*1000.

```

```

55. UNITS(1)=ENTNY(2,3)
56. UNITS(2)=ENTNY(3,3)
57. RETURN
58. C
59. IF CODE NOT FOUND. RETURN WITH 'UNITS ERROR' IN 'UNITS.'.
60. 3 UNITS(1)=UNITS
61. UNITS(2)=ENRNO
62. PRINT IQLALCODE
63. IQL FORMAT: ERROR--UNIT CODE '.13.' NOT IN ANGLES TABLE.'.
64. RETURN
65. END

```


QLLT,L N,AREACK,,AREACK

1. SUBROUTINE AREACK (INLAT,ELON,SLAT,ELON,ILAT,ILON,FLAT,FLON,,)

2. C

3. C

4. IMPLICIT INTEGER (A-Z)

5. C

6. IF INLAT .EQ. 0 .AND. SLAT .EQ. 0 .AND.

7. ELON .EQ. 0 .AND. ELON .EQ. 0) RETURN

8. IF (ILAT .LT. SLAT .OR. FLAT .GT. NLAT) RETURN 9

9. IF (WEST(ELON,ILON) .EQ. ILON .OR.

10. EAST(ELON,FLON) .EQ. FLON) RETURN 9

11. RETURN

12. END

DELTA, L H, ASUNPK, ASUNPK

1: SUBROUTINE ASUNPK(STABFM, TYPE)
2: IMPLICIT INTEGER(A-Z)

3: C
4: C
5: C

6: COMMON /ASTATN/SANO, SANRUN, SAILAT, SAILON, SAIDAT, SAIFLAT, SAFLON,
7: SAFDAT, SASAT(2), PNATAS, PENVS, SAIFLG

8: C
9: C
10: C

11: DIMENSION STABFR(1)

12: C
13: C

14: C UNPACK TYPE, STATION NUMBER, AND NUMBER OF RUNS.

15: TYPE=FLD(15,3), STABFR(1)

16: SANO=FLD(16,18), STABFR(2)

17: SANRUN=FLD(17,19), STABFR(2)

18: C
19: C

20: C UNPACK STATION INITIAL AND FINAL POINT BOUNDS.

21: CALL UPKREAL('SAILAT, SAILON, SAIDAT, 0', 1

22: C SAFLAT, SAFLON, SAFDAT, 0', 3, STABFR(3))

23: C
24: C

25: C UNPACK STATION STATUS BITS.

26: SASAT(1)=STABFR(7)

27: SASAT(2)=STABFR(8)

28: C
29: C

30: PNATAS=STABFR(9)

31: C
32: C

33: PENVS=STABFR(10)

34: RETURN

35: END

DELTA N.AUNPCK, AUNPCK

1. SUBROUTINE AUNPCK (BFR, IPTN)

5. IMPLICIT INTEGER (A-Z)

8. COMMON /RUN /PORA, RELAT, RINS, NFLAT, RFNS, RILON, RIEL, RFELON, RFEN,
 9. RIMO, RIDAY, RIYR, KITIM, NIZON, RFNG, RFDAY, RFYR, MFTIM,
 10. AFZON, NAVCOD, DTASRC, SYSSRC, SRCRTYP, RCRTYP,
 11. CLAS, DESCR, WAVDIR, WAVMT, MTUNIT, WAVPKD, PHDUNT,
 12. SEASTA, NDOLR, ENDVEL, VELUNT, SALUTR, SWLHT,
 13. SHTUNT, SALPRD, SPDUNT, WEATHN, BOTDPH, UPHUNT, BTNSLP,
 14. BATHY, INFO(9), RSTAT(2), RUNNO
 15. REAL WAVDIR, WAVMT, WAVPKD, SEASTA, NDOLR, ENDVEL, SWLHT, R,
 16. SWLHT, SALPRD, WEATHN, BOTDPH, BTNSLP

20. COMMON /CPV /NCON, CON(4,30), MPAR, PAR(30,30), NPAR, IVAR(3,30),
 21. HDATS, MKO, S(50), NDATA, VALNG(2,30,50), DATA(30000)
 22. DIMENSION ICON(4,30), IPAR(30,30), IVALNG(2,30,50)
 23. EQUIVALENCE (CON(1,1), ICON(1,1)), (PAR(1,1), IPAR(1,1)),
 24. (VALNG(1,1), IVALNG(1,1))

25. REAL CON, PAR, DATA, VALNG

28. DIMENSION BFR(1)
 29. REAL BFR

32. ISUB = IPTN

34. UNPACK RUNNO

36. RUNNO = FLD(0,18,BFR(1SUB))

38. UNPACK NAVCOD, DTASRC, CLAS, DESCR

40. ISUB = ISUB + 1
 41. NAVCOD = FLD(0,9,BFR(1SUB))
 42. DTASRC = FLD(0,9,BFR(1SUB))
 43. CLAS = FLD(18,9,BFR(1SUB))
 44. DESCR = FLD(27,9,BFR(1SUB))

46. UNPACK SYSSRC, SYSSRC, SRCRTYP, RCRTYP

48. ISUB = ISUB + 1
 49. SYSSRC = FLD(0,9,BFR(1SUB))
 50. SYSSRC = FLD(0,9,BFR(1SUB))
 51. SRCRTYP = FLD(18,9,BFR(1SUB))
 52. RCRTYP = FLD(27,9,BFR(1SUB))

54. UNPACK AREA/TIME

```

55. C
56. ISUB = ISUB + 1
57. CALL UPKREA (PORA, MILAT, NILON, MIDAY, RITIM, NIZON,
58. NPLAT, NPELON, MPDAY, MPTIM, NPZON, BFM(1SUB))
59.
60. C UNPACK STATUS BITS
61. C
62. ISUB = ISUB + 4
63. RSTAT(1) = FLD(3,36,BFM(1SUB))
64. RSTAT(2) = FLD(3,36,BFM(1SUB+1))
65. C
66. C THIS ENDS THE UNPACKING OF DATA INTO COMMON /RUN/. FROM HERE
67. C ON THE DATA WILL BE UNPACKED INTO /CPV/.
68. C
69. C UNPACK NCON, NPAR, NVAR, NOATS
70. C
71. ISUB = ISUB + 3
72. NCON = FLD( 0,9,BFM(1SUB))
73. NPAR = FLD( 9,9,BFM(1SUB))
74. NVAR = FLD(18,9,BFM(1SUB))
75. NOATS = FLD(27,9,BFM(1SUB))
76. C
77. C UNPACK NAME CODES & PROCESS CODES FOR ALL CONSTANTS
78. C
79. C IF (NCON.EQ.0) GO TO 45
80. DO 30 I=1,NCON,2
81. ISUB = ISUB + 1
82. ICON(1,I) = FLD( 3,8,BFM(1SUB))
83. ICON(4,I) = FLD( 8,13,BFM(1SUB))
84. J = I+1
85. IF (J.GT. NCON) GO TO 30
86. ICON(1,J) = FLD(18,8,BFM(1SUB))
87. ICON(4,J) = FLD(26,13,BFM(1SUB))
88. 30 CONTINUE
89. C UNPACK VALUES FOR CONSTANTS
90. C
91. C DO 40 I=1,NCON
92. ISUB = ISUB + 1
93. CON(3,I) = BFM(1SUB)
94. 40 CONTINUE
95. C UNPACK DATA FOR ALL PARAMETERS
96. C
97. C 45 IF (NPAR.EQ.0) GO TO 65
98. DO 60 I=1,NPAR
99. C UNPACK FOR PARAMETER(I)
100. C
101. C ISUB = ISUB + 1
102. IPAR(1,I) = FLD( 0,9,BFM(1SUB))
103. IPAR(2,I) = FLD( 9,8,BFM(1SUB))
104. IPAR(4,I) = FLD(17,13,BFM(1SUB))
105. C UNPACK ALL VALUES FOR PARAMETER(I)
106. C
107. C JJ = IPAR(1,I)
108. C
109. C
110. C
111. C

```

```

112. DO 50 J=1,JJ
113.   ISUB = ISUB + 1
114.   PAR(4,J,1) = BFR(ISUB)
115.   50 CONTINUE
116.   C
117.   60 CONTINUE
118.   C
119.   C
120.   65 IF (NVAR .EQ. 0) GO TO 101
121.   DO 70 I=1,NVAR,2
122.     ISUB = ISUB + 1
123.     IVAR(1,1) = FLD(1,0,BFR(ISUB))
124.     IVAR(3,1) = FLD(1,0,BFR(ISUB))
125.     J = 1+1
126.     IF (J .GT. NVAR) GO TO 70
127.     IVAR(1,J) = FLD(1,0,BFR(ISUB))
128.     IVAR(3,J) = FLD(1,0,BFR(ISUB))
129.   70 CONTINUE
130.   C
131.   C
132.   DO 75 I=1,NVARS,2
133.     ISUB = ISUB + 1
134.     NROWS(I) = FLD(0,10,BFR(ISUB))
135.     J = 1+1
136.     IF (J .GT. NVARS) GO TO 75
137.     NROWS(J) = FLD(0,10,BFR(ISUB))
138.   75 CONTINUE
139.   C
140.   C
141.   C
142.   DO 100 K=1,NVARS
143.     C
144.     DO 90 J=1,NVAR
145.       C
146.       DO 80 I=1,2
147.         ISUB = ISUB + 1
148.         VALRG(I,J,K) = BFR(ISUB)
149.         80 CONTINUE
150.       C
151.       90 CONTINUE
152.     C
153.     100 CONTINUE
154.     C
155.     C
156.     101 IPTR = ISUB + 1
157.     C
158.     C
159.     C
160.     C
161.     C
162.     C
163.     C
164.     C
165.     C
166.     C
167.     C
168.     C
169.     C
170.     C
171.     C
172.     C
173.     C
174.     C
175.     C
176.     C
177.     C
178.     C
179.     C
180.     C
181.     C
182.     C
183.     C
184.     C
185.     C
186.     C
187.     C
188.     C
189.     C
190.     C
191.     C
192.     C
193.     C
194.     C
195.     C
196.     C
197.     C
198.     C
199.     C
200.     C
201.     C
202.     C
203.     C
204.     C
205.     C
206.     C
207.     C
208.     C
209.     C
210.     C
211.     C
212.     C
213.     C
214.     C
215.     C
216.     C
217.     C
218.     C
219.     C
220.     C
221.     C
222.     C
223.     C
224.     C
225.     C
226.     C
227.     C
228.     C
229.     C
230.     C
231.     C
232.     C
233.     C
234.     C
235.     C
236.     C
237.     C
238.     C
239.     C
240.     C
241.     C
242.     C
243.     C
244.     C
245.     C
246.     C
247.     C
248.     C
249.     C
250.     C
251.     C
252.     C
253.     C
254.     C
255.     C
256.     C
257.     C
258.     C
259.     C
260.     C
261.     C
262.     C
263.     C
264.     C
265.     C
266.     C
267.     C
268.     C
269.     C
270.     C
271.     C
272.     C
273.     C
274.     C
275.     C
276.     C
277.     C
278.     C
279.     C
280.     C
281.     C
282.     C
283.     C
284.     C
285.     C
286.     C
287.     C
288.     C
289.     C
290.     C
291.     C
292.     C
293.     C
294.     C
295.     C
296.     C
297.     C
298.     C
299.     C
300.     C
301.     C
302.     C
303.     C
304.     C
305.     C
306.     C
307.     C
308.     C
309.     C
310.     C
311.     C
312.     C
313.     C
314.     C
315.     C
316.     C
317.     C
318.     C
319.     C
320.     C
321.     C
322.     C
323.     C
324.     C
325.     C
326.     C
327.     C
328.     C
329.     C
330.     C
331.     C
332.     C
333.     C
334.     C
335.     C
336.     C
337.     C
338.     C
339.     C
340.     C
341.     C
342.     C
343.     C
344.     C
345.     C
346.     C
347.     C
348.     C
349.     C
350.     C
351.     C
352.     C
353.     C
354.     C
355.     C
356.     C
357.     C
358.     C
359.     C
360.     C
361.     C
362.     C
363.     C
364.     C
365.     C
366.     C
367.     C
368.     C
369.     C
370.     C
371.     C
372.     C
373.     C
374.     C
375.     C
376.     C
377.     C
378.     C
379.     C
380.     C
381.     C
382.     C
383.     C
384.     C
385.     C
386.     C
387.     C
388.     C
389.     C
390.     C
391.     C
392.     C
393.     C
394.     C
395.     C
396.     C
397.     C
398.     C
399.     C
400.     C
401.     C
402.     C
403.     C
404.     C
405.     C
406.     C
407.     C
408.     C
409.     C
410.     C
411.     C
412.     C
413.     C
414.     C
415.     C
416.     C
417.     C
418.     C
419.     C
420.     C
421.     C
422.     C
423.     C
424.     C
425.     C
426.     C
427.     C
428.     C
429.     C
430.     C
431.     C
432.     C
433.     C
434.     C
435.     C
436.     C
437.     C
438.     C
439.     C
440.     C
441.     C
442.     C
443.     C
444.     C
445.     C
446.     C
447.     C
448.     C
449.     C
450.     C
451.     C
452.     C
453.     C
454.     C
455.     C
456.     C
457.     C
458.     C
459.     C
460.     C
461.     C
462.     C
463.     C
464.     C
465.     C
466.     C
467.     C
468.     C
469.     C
470.     C
471.     C
472.     C
473.     C
474.     C
475.     C
476.     C
477.     C
478.     C
479.     C
480.     C
481.     C
482.     C
483.     C
484.     C
485.     C
486.     C
487.     C
488.     C
489.     C
490.     C
491.     C
492.     C
493.     C
494.     C
495.     C
496.     C
497.     C
498.     C
499.     C
500.     C
501.     C
502.     C
503.     C
504.     C
505.     C
506.     C
507.     C
508.     C
509.     C
510.     C
511.     C
512.     C
513.     C
514.     C
515.     C
516.     C
517.     C
518.     C
519.     C
520.     C
521.     C
522.     C
523.     C
524.     C
525.     C
526.     C
527.     C
528.     C
529.     C
530.     C
531.     C
532.     C
533.     C
534.     C
535.     C
536.     C
537.     C
538.     C
539.     C
540.     C
541.     C
542.     C
543.     C
544.     C
545.     C
546.     C
547.     C
548.     C
549.     C
550.     C
551.     C
552.     C
553.     C
554.     C
555.     C
556.     C
557.     C
558.     C
559.     C
560.     C
561.     C
562.     C
563.     C
564.     C
565.     C
566.     C
567.     C
568.     C
569.     C
570.     C
571.     C
572.     C
573.     C
574.     C
575.     C
576.     C
577.     C
578.     C
579.     C
580.     C
581.     C
582.     C
583.     C
584.     C
585.     C
586.     C
587.     C
588.     C
589.     C
590.     C
591.     C
592.     C
593.     C
594.     C
595.     C
596.     C
597.     C
598.     C
599.     C
600.     C
601.     C
602.     C
603.     C
604.     C
605.     C
606.     C
607.     C
608.     C
609.     C
610.     C
611.     C
612.     C
613.     C
614.     C
615.     C
616.     C
617.     C
618.     C
619.     C
620.     C
621.     C
622.     C
623.     C
624.     C
625.     C
626.     C
627.     C
628.     C
629.     C
630.     C
631.     C
632.     C
633.     C
634.     C
635.     C
636.     C
637.     C
638.     C
639.     C
640.     C
641.     C
642.     C
643.     C
644.     C
645.     C
646.     C
647.     C
648.     C
649.     C
650.     C
651.     C
652.     C
653.     C
654.     C
655.     C
656.     C
657.     C
658.     C
659.     C
660.     C
661.     C
662.     C
663.     C
664.     C
665.     C
666.     C
667.     C
668.     C
669.     C
670.     C
671.     C
672.     C
673.     C
674.     C
675.     C
676.     C
677.     C
678.     C
679.     C
680.     C
681.     C
682.     C
683.     C
684.     C
685.     C
686.     C
687.     C
688.     C
689.     C
690.     C
691.     C
692.     C
693.     C
694.     C
695.     C
696.     C
697.     C
698.     C
699.     C
700.     C
701.     C
702.     C
703.     C
704.     C
705.     C
706.     C
707.     C
708.     C
709.     C
710.     C
711.     C
712.     C
713.     C
714.     C
715.     C
716.     C
717.     C
718.     C
719.     C
720.     C
721.     C
722.     C
723.     C
724.     C
725.     C
726.     C
727.     C
728.     C
729.     C
730.     C
731.     C
732.     C
733.     C
734.     C
735.     C
736.     C
737.     C
738.     C
739.     C
740.     C
741.     C
742.     C
743.     C
744.     C
745.     C
746.     C
747.     C
748.     C
749.     C
750.     C
751.     C
752.     C
753.     C
754.     C
755.     C
756.     C
757.     C
758.     C
759.     C
760.     C
761.     C
762.     C
763.     C
764.     C
765.     C
766.     C
767.     C
768.     C
769.     C
770.     C
771.     C
772.     C
773.     C
774.     C
775.     C
776.     C
777.     C
778.     C
779.     C
780.     C
781.     C
782.     C
783.     C
784.     C
785.     C
786.     C
787.     C
788.     C
789.     C
790.     C
791.     C
792.     C
793.     C
794.     C
795.     C
796.     C
797.     C
798.     C
799.     C
800.     C
801.     C
802.     C
803.     C
804.     C
805.     C
806.     C
807.     C
808.     C
809.     C
810.     C
811.     C
812.     C
813.     C
814.     C
815.     C
816.     C
817.     C
818.     C
819.     C
820.     C
821.     C
822.     C
823.     C
824.     C
825.     C
826.     C
827.     C
828.     C
829.     C
830.     C
831.     C
832.     C
833.     C
834.     C
835.     C
836.     C
837.     C
838.     C
839.     C
840.     C
841.     C
842.     C
843.     C
844.     C
845.     C
846.     C
847.     C
848.     C
849.     C
850.     C
851.     C
852.     C
853.     C
854.     C
855.     C
856.     C
857.     C
858.     C
859.     C
860.     C
861.     C
862.     C
863.     C
864.     C
865.     C
866.     C
867.     C
868.     C
869.     C
870.     C
871.     C
872.     C
873.     C
874.     C
875.     C
876.     C
877.     C
878.     C
879.     C
880.     C
881.     C
882.     C
883.     C
884.     C
885.     C
886.     C
887.     C
888.     C
889.     C
890.     C
891.     C
892.     C
893.     C
894.     C
895.     C
896.     C
897.     C
898.     C
899.     C
900.     C
901.     C
902.     C
903.     C
904.     C
905.     C
906.     C
907.     C
908.     C
909.     C
910.     C
911.     C
912.     C
913.     C
914.     C
915.     C
916.     C
917.     C
918.     C
919.     C
920.     C
921.     C
922.     C
923.     C
924.     C
925.     C
926.     C
927.     C
928.     C
929.     C
930.     C
931.     C
932.     C
933.     C
934.     C
935.     C
936.     C
937.     C
938.     C
939.     C
940.     C
941.     C
942.     C
943.     C
944.     C
945.     C
946.     C
947.     C
948.     C
949.     C
950.     C
951.     C
952.     C
953.     C
954.     C
955.     C
956.     C
957.     C
958.     C
959.     C
960.     C
961.     C
962.     C
963.     C
964.     C
965.     C
966.     C
967.     C
968.     C
969.     C
970.     C
971.     C
972.     C
973.     C
974.     C
975.     C
976.     C
977.     C
978.     C
979.     C
980.     C
981.     C
982.     C
983.     C
984.     C
985.     C
986.     C
987.     C
988.     C
989.     C
990.     C
991.     C
992.     C
993.     C
994.     C
995.     C
996.     C
997.     C
998.     C
999.     C
1000.    C

```


PFOR,SA N,CHECK,=CHECK

```

1. SUBROUTINE LMECK (VALUE,NAME,UNITS)
2. C
3. C CHECK CHECKS TO SEE IF THE NAME CODE 'NAME' HAS BEEN
4. C ENTERED IN EITHER THE ACOUSTIC PARAMETER OR ENV PARAMETER
5. C ON THE RETRIEVAL REQUEST. IF IT HAS, IT CONVERTS THE VALUE
6. C 'VALUE' TO THE REQUESTED UNITS & PUTS THE NAME OF THE UNITS
7. C IN 'UNITS'. IF NOT, IT DOES NOT CHANGE THE VALUE OF 'VALUE'
8. C 6 PUTS THE NAME OF THE STORED UNITS IN 'UNITS'.
9. C
10. C
11. C IMPLICIT INTEGER (A-Z)
12. C
13. C
14. C INCLUDE RETNEVALIST
15. C
16. C
17. C
18. C MEAL VALUE
19. C DIMENSION UNITS(2)
20. C
21. C CHECK ACOUSTICS
22. C
23. C TEMP=0
24. C IF (INACPAR.EQ. C) GO TO 10
25. C DO 7 I=1,NACPAR
26. C IF (ACPAR(I,1).NE. NAME) GO TO 7
27. C TEMP = -ACPAR(I,1)
28. C GO TO 30
29. C 7 CONTINUE
30. C
31. C CHECK ENVIRONMENTAL
32. C
33. C 10 IF (INENPAR.EQ. D) GO TO 30
34. C TEMP = 0
35. C DO 20 I=1,NENPAR
36. C IF (ENPAR(I,1).NE. NAME) GO TO 20
37. C TEMP = -ENPAR(I,1)
38. C GO TO 30
39. C 20 CONTINUE
40. C
41. C 30 CALL CONVRT (VALUE,NAME,TEMP,UNITS)
42. C
43. C 40 RETURN
44. C END

```

1.	C	SUBROUTINE CLASS (NOSEL,SEL,MASK,REQUEST,INAME,INBR,FIRST,BACKUP, ICLSREQ)
2.		
3.		
4.	C	THIS ROUTINE PROVIDES FOR SIGNING ON AND THE INPUT OF THE USER'S DESIRED SECURITY LEVELS.
5.	C	

```

7.  PARAMETER UNIT = 12
8.  IMPLICIT INTEGER (A-Z)
9.  LOGICAL PASS, SECURE, FIRST, BACKUP, CLASREQ, NUMCK, PASSCK
10. REAL WORDS (10)
11. DIMENSION SEX(1), NAME(5), TNAME(5), REQUEST(3), CHAR(61), MASK(3),
12. ILEVEL(3), REJECT(3), REJ(10)
13. DATA A/00000505050505/, Z/037050505050505/, CLASCK/052777777777777/,
14. IALL/0777777777777777/, ALLI/0777777777777777/, NUMCK/0.FALSE/,
15. ZPASSCK/0.FALSE/

```

17.	C	IF BACKING UP, BRANCH TO APPROPRIATE QUESTION
18.	C	
19.		IF (NOT BACKUP) GO TO 1
20.		IF (CLREQ) GO TO 220
21.		IF (FIRST) GO TO 8

```

23. C CLASSIFIED DATA REQUEST QUESTION
24. C
25. 1 WRITE (A,2)
26. 2 FORMAT (1000 YOU INTEND TO RETRIEVE ANY CLASSIFIED DATA?)
27. READ (B,2,EQ=1,ERR=5) ANSWER
28. 3 FORMAT (A3)
29. WRITE (A,3) ANSWER
30. 4 FORMAT (3X,A3)
31. 5 . . .
32. FLD (0.6,W) = FLD (0.6,ANSWER)
33. IF (W.EQ.1) GO TO 15
34. IF (W.EQ. .008.W.EQ.W) GO TO 7

```

32.	C	ERROR
36.	C	
37.	C	
38.	S	WRITE (0.6)
39.	6	INVALID RESPONSE TO YES OR NO QUESTION ****)
40.		FORMAT 1: *** GO TO 1

```

42. C UNCLASSIFIED REQUEST (ENTER NAME IF FIRST PASS)
43. C
44. 7 CLSREQ = .FALSE.
45. IF (.NOT.FIRST) GO TO 14
46. 8 WRITE (6,9)
47. 9 FORMAT (1,ENTER USER NAME:)
48. 10 INDR = -1
49. READ (5,10,END=1,ERR=12) (TNAME(1),I=1,5)
50. 11 FORMAT (5A6)
51. WRITE (6,11) (TNAME(1),I=1,5)
52. 12 FORMAT (3X,5A6)
53. 13 GO TO 14
54. C

```

```

55. C ERROR
56. C
57. 12 WRITE (6,13)
58. 13 FORMAT (I, '... READ FORMAT ERROR ....')
59. 60 TO 6
60. C
61. C
62. 14 NOSEX = 1
63. 15 SEX(1) = 1
64. REQUEST(1) = 0
65. REQUEST(2) = 0
66. REQUEST(3) = 0
67. 16 FLD(1,1,REQUEST(1)) = 1
68. RETURN
69. C
70. C
71. 15 CLSREQ = .TRUE.
72. IF (NUNCK) 50 TO 220
73. 16 COUNT = 0
74. 18 COUNT = COUNT + 1
75. IF (COUNT.E.3) 60 TO 30
76. 19 WRITE (6,20)
77. 20 FORMAT (10000 USER NAME AND/OR ID WAS ENTERED INCORRECTLY IN THREE
78. 1 ATTEMPTS. / SEX PROGRAM TERMINATES ....)
79. STOP ERROR
80. C
81. 30 WRITE (6,40)
82. 40 FORMAT (10000 ENTER USER NAME, USER ID)
83. READ (5,50,END=1,ERR=95) (CHAR(1),1=1,80)
84. 50 FORMAT (80A1)
85. 60 WRITE (6,60) (CHAR(1),1=1,80)
86. 60 FORMAT (13X,60A1)
87. CALL IDNAME (CHAR,10,NAME,FLAG)
88. IF (FLAG.EQ.0) 60 TO 120
89. 60 TO 30
90. C
91. C ERROR
92. C
93. 95 WRITE (6,100)
94. 100 FORMAT (I, '... ERROR ENTERING USER NAME OR ID ....')
95. 60 TO 30
96. C
97. C
98. 120 REWIND UNIT
99. N = 0
100. 120 N = N + 1
101. READ (UNIT,140,END=180,ERR=200) TNR,(TNAME(1),1=1,5),
102. 1 (LEVELS(1),1=1,3)
103. 140 FORMAT (6A4,30I2)
104. IF (10.NE.TNR) 60 TO 130
105. DO 180 I = 1,5
106. 150 IF (NAME(1).NE.TNAME(1)) 60 TO 160
107. 60 TO 220
108. C
109. C ERROR CONDITIONS
110. C
111. 160 WRITE (6,170)

```

```

112. 170 FORMAT (' *** USER NAME DOES NOT MATCH NAME IN TABLE ****')
113. GO TO 18
114. 180 WRITE (4,190) ID
115. 190 FORMAT (' *** USER ID ', A4, ' NOT FOUND IN TABLE ****')
116. GO TO 18
117. 200 WRITE (4,210) N
118. 210 FORMAT (' *** ERROR READING RECORD', 14, ' OF MASTER USER FILE. ')
119. 1. CONTACT BRANCH MANAGER REGARDING ERROR ****
120. GO TO 130
121. C
122. C ENTER SECURITY LEVELS
123. C
124. 220 WRITE (4,230)
125. 230 FORMAT (' ENTER SECURITY LEVEL(S)')
126. REQUEST(1) = 0
127. REQUEST(2) = 0
128. REQUEST(3) = 0
129. READ (5,50,END=255,ERR=240) (CHAR(1),1=1,50)
130. CALL FREEFD (CHAR,WORDS,SEX,MOSEX,10,5240)
131. GO TO 240
132. C
133. C ERROR
134. C
135. 240 WRITE (4,250)
136. 250 FORMAT (' *** ERROR ENTERING SECURITY LEVELS ****')
137. GO TO 220
138. C
139. C BACKUP TO LAST QUESTION
140. C
141. 255 IF (.NOT.NUMCK) GO TO 14
142. GO TO 1
143. C
144. 260 SECURE = .FALSE.
145. IF (MOSEX.EQ.0) GO TO 260
146. DO 270 I = 1,MOSEX
147. ISEX = IABS(IX(1))
148. IF (ISEX.GT.99) GO TO 360
149. IF (ISEX.NE.1) SECURE = .TRUE.
150. WORD = ISEX/34 + 1
151. BIT = MOD(ISEX,34)
152. FLD (BIT,1,REQUEST(WORD)) = 1
153. IF (ISEX.LE.0) GO TO 270
154. FLD (1,1,REQUEST(1)) = 1
155. IF (ISEX.LE.0) FLD (2,1,REQUEST(1)) = 1
156. CONTINUE
157. NUMCK = .TRUE.
158. IF (SECURE) GO TO 320
159. C
160. C UNCLASSIFIED ONLY
161. C
162. 280 FLD (1,1,REQUEST(1)) = 1
163. NUMCK = .TRUE.
164. WRITE (4,290)
165. 290 FORMAT (' YOU HAVE REQUESTED UNCLASSIFIED DATA ONLY')
166. GO TO 360
167. C
168. C ERROR - LEVEL OUT OF RANGE

```



```

169. C
170. 30C WRITE (4,310) SEX(1)
171. 31C FORMAT (1,000 SECURITY LEVEL,15,' OUT OF RANGE .....')
172. GO TO 220
173. C
174. C
175. 32C IF (FLD(0,1,REQUEST(1))-NE,1) GO TO 340
176. WRITE (4,330)
177. 33C FORMAT (1, YOUR REQUEST IS FOR ALL SECURITY CLASSIFICATIONS')
178. FLD (0,30,REQUEST(1)) = FLD (0,30,ALL)
179. FLD (0,30,REQUEST(2)) = FLD (0,30,ALL)
180. FLD (0,30,REQUEST(3)) = FLD (0,30,ALL)
181. GO TO 380
182. C
183. C PRINT REQUESTED LEVELS
184. C
185. 340 WRITE (4,350)
186. 350 FORMAT (1, YOUR REQUEST IS FOR THE FOLLOWING SECURITY LEVEL(S):')
187. DO 370 I = 1,99
188. WREQ = 1/234 + I
189. BIT = MOD(I,36)
190. IF (FLD(BIT,1,REQUEST(BIT))-EQ,0) GO TO 370
191. CALL NAMEIT (2,1,NAME,WR)
192. WRITE (4,360) (NAME(J),J=1,WR)
193. 360 FORMAT (9X,5A4)
194. 370 CONTINUE
195. 380 IF (FLD(0,1,LEVELS(1))-NE,1) GO TO 390
196. C
197. C STORE 1'S IN STATUS BITS
198. C
199. FLD (0,30,LEVELS(1)) = FLD (0,30,ALL)
200. FLD (0,30,LEVELS(2)) = FLD (0,30,ALL)
201. FLD (0,30,LEVELS(3)) = FLD (0,30,ALL)
202. C
203. C
204. 390 DO 400 I = 1,3
205. MASK(I) = AND (REQUEST(I),LEVELS(I))
206. 400 REJECT(I) = 10R (REQUEST(I),MASK(I))
207. C
208. C PRINT REJECTED LEVELS (CHECK IF NO AUTHORIZED LEVELS)
209. C
210. IF (REJECT(1)-EQ,0,AND,REJECT(2)-EQ,0,AND,REJECT(3)-EQ,0) GO TO 460
211. IF (MASK(1)-NE,0,OR,MASK(2)-NE,0,OR,MASK(3)-NE,0) GO TO 420
212. WRITE (4,410)
213. 410 FORMAT (1, ALL REQUESTED LEVELS ARE UNAUTHORIZED,')
214. 1, CONTACT BRANCH MANAGER FOR QUESTIONS REGARDING REJECTED LEVELS,')
215. GO TO 220
216. 420 WRITE (4,430)
217. 430 FORMAT (1, THE FOLLOWING REQUESTED LEVELS ARE UNAUTHORIZED, OTHERS
218. 1, ACCEPTED,') / 1X, 1, CONTACT BRANCH MANAGER FOR QUESTIONS REGARDING R
219. 2, REJECTED LEVELS,')
220. N = 0
221. DO 450 I = 1,99
222. WREQ = 1/236 + I
223. BIT = MOD(I,36)
224. IF (FLD(BIT,1,REJECT(BIT))-EQ,0) GO TO 460
225. N = N + 1

```

```

226. REJIN = 1
227. IF (N-LT,10) GO TO 450
228. WRITE (6,940) (REJ(J),J=1,M)
229. 440 FORMAT (5X,10I12,' ')
230. N = 0
231. 450 CONTINUE
232. IF (N-6T,C) WRITE (6,940) (REJ(J),J=1,M)
233. C
234. C CHECK FOR CLASSIFIED
235. C
236. 460 IF (AND(MASK(1),CLASSCT.EQ.0.AND.(MASK(2),ALL).EQ.0.AND.
237. 1AND(MASK(3),ALL).EQ.0) GO TO 590
238. C
239. C
240. 470 IF (PASSCT) GO TO 580
241. WRITE (6,500)
242. 480 FORMAT (10YOU HAVE REQUESTED CLASSIFIED DATA.)
243. COUNT = 0
244. 490 COUNT = COUNT + 1
245. IF (COUNT-3) GO TO 510
246. WRITE (6,500)
247. 500 FORMAT (1'0000 PASSWORD WAS ENTERED INCORRECTLY IN THREE ATTEMPTS.'
248. 1/ 5X, 'PROGRAM TERMINATES 0000')
249. STOP ERROR
250. 510 WRITE (6,520)
251. 520 FORMAT (1' ENTER PASSWORD.)
252. READ (5,530,END=220,ERR=550) PASSWD
253. 530 FORMAT (A1)
254. C
255. C
256. IF (PASS(PASSWD)) GO TO 570
257. CALL OVERLY
258. C
259. C ERROR CONDITION
260. C
261. WRITE (6,540) PASSED
262. 540 FORMAT (1' PASSWORD 'A0.' IS INCORRECT'/)
263. GO TO 490
264. 550 WRITE (6,560)
265. 560 FORMAT (1'000 ERROR ENTERING PASSWORD 000'/)
266. GO TO 510
267. C
268. 570 CALL OVERLY
269. PASSCK = .TRUE.
270. C
271. C
272. 580 IF (FLC(10,1,MASK(1)),NE.1) GO TO 590
273. NOSEX = 1
274. SEX(1) = C
275. RETURN
276. 590 NOSEX = 0
277. DO 600 I = 1,99
278. WORD = 1/36 + I
279. BIT = MOD(I,30)
280. IF (FLC(10,1,MASK(MORD)),EQ.0) GO TO 600
281. NOSEX = NOSEX + 1
282. SEX(MORD) = I

```

23. CONTINUE
24. RETURN
25. END

OFOR,3# N.CONVRY,CONVRT

```

1. SUBROUTINE CONVERTVALUE,INDXS,UNICODE,UNITS)
2. INTEGER TABLE,UNICODE
3. C
4. C SET UP TABLE TO MAP VARIABLE CODE TO PROPER TABLE NUMBER.
5. C
6. C DIMENSION TABLE(144),UNITS(14)
7. C DATA NULL /057637777777/
8. C DATA (TABLE(1),1=1,72 1/00401,00401,07712,07712,00010,02204,00401,
9. C 00401,00010,00401,00401,00401,00010,00501,00010,
10. C 03000,07712,07712,00010,00401,02204,00010,
11. C 00401,00401,00010,00010,02204,00000,03000,
12. C 07712,03000,03200,02204,03000,03000,
13. C 3000/
14. C
15. C DATA (TABLE(1),1=73,144)/00000,00401,02204,00000,03000,02204,00000,
16. C 00401,02204,00000,00000,00000,00401,00401,
17. C 00401,03000,04300,02900,00000,00301,00301,
18. C 00517,04209,04209,04209,04209,04209,00601,00501,
19. C 1400/
20. C
21. C UNITS(1)=UNITLE*
22. C UNITS(2)=UNITLE*
23. C UNITS(3)=UNITLE*
24. C UNITS(4)=UNITLE*
25. C UNITS(5)=UNITLE*
26. C
27. C IF (FLO(0,30,VALUE).EQ.FLO(0,30,ANULL)) RETURN
28. C
29. C
30. C
31. C MAP CODES 1-72 AND 101-172 TO CONTINUOUS TABLE POSITIONS 1-144.
32. C
33. C INDEX=INDXS
34. C IF (INDEX.GT.100) INDEX=INDEX-28
35. C
36. C IF INDEX IS OUT OF RANGE, RETURN VALUE.
37. C
38. C
39. C IF (INDEX.GE.1.AND.INDEX.LE.144) GO TO 101
40. C PRINT 102,INDXS
41. C UNITS(1)=UNITLE*
42. C UNITS(2)=UNITLE*
43. C RETURN
44. C
45. C CALCULATE LABEL FOR APPROPRIATE UNIT-CODE TABLE SUBROUTINE.
46. C
47. C 101 IF (TABLE(INDEX).EQ.0) RETURN
48. C ITHPCD = TABLE(INDEX) / 100
49. C INDEX = TABLE(INDEX) - ITHPCD * 100
50. C IF (UNICODE.EQ.C) UNITLE = ITHPCD
51. C
52. C BRANCH TO CORRECT UNIT-CODE TABLE SUBROUTINE.
53. C
54. C GO TO 11,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,1,INDEX

```


55.	C	
56.	C	'LINEAR' TABLE.
57.	C	
58.		1 CALL LINEAR(VALUE,UNICODE,UNITS)
59.		RETURN
60.	C	
61.	C	'PLANAR' TABLE.
62.	C	
63.		2 CALL PLANAR(VALUE,UNICODE,UNITS)
64.		RETURN
65.	C	
66.	C	'VOLUME' TABLE.
67.	C	
68.		3 CALL VOLUME(VALUE,UNICODE,UNITS)
69.		RETURN
70.	C	
71.	C	'SHORT TIME' TABLE.
72.	C	
73.		4 CALL SMRTIME(VALUE,UNICODE,UNITS)
74.		RETURN
75.	C	
76.	C	'LONG TIME' TABLE.
77.	C	
78.		5 CALL LONGTIME(VALUE,UNICODE,UNITS)
79.		RETURN
80.	C	
81.	C	'VELOCITY' TABLE.
82.	C	
83.		6 CALL SPEEDS(VALUE,UNICODE,UNITS)
84.		RETURN
85.	C	
86.	C	'TEMPERATURE' TABLE.
87.	C	
88.		7 CALL IMPTUR(VALUE,UNICODE,UNITS)
89.		RETURN
90.	C	
91.	C	'ANGLE' TABLE.
92.	C	
93.		8 CALL ANGLES(VALUE,UNICODE,UNITS)
94.		RETURN
95.	C	
96.	C	'PER CENT' TABLE.
97.	C	
98.		9 CALL PERCENT(VALUE,UNICODE,UNITS)
99.		RETURN
100.	C	
101.	C	'PRESSURE' TABLE.
102.	C	
103.		10 CALL PRESSR(VALUE,UNICODE,UNITS)
104.		RETURN
105.	C	
106.	C	'ACOUSTIC PRESSURE' TABLE.
107.	C	
108.		11 CALL ACPRES(VALUE,UNICODE,UNITS)
109.		RETURN
110.	C	
111.	C	'FREQUENCY' TABLE.

```

112. C
113. 12 CALL CYCLES(VALUE,UNICODE,UNITS)
114. RETURN
115. C
116. 'ENERGY' TABLE.
117. C
118. 13 CALL ENERGY(VALUE,UNICODE,UNITS)
119. RETURN
120. C
121. 'POWER' TABLE.
122. C
123. 14 CALL POWER(VALUE,UNICODE,UNITS)
124. RETURN
125. C
126. 'INTENSITY' TABLE.
127. C
128. 15 CALL INTENS(VALUE,UNICODE,UNITS)
129. RETURN
130. C
131. 'PRESSURE LEVEL' TABLE.
132. C
133. 16 CALL PRESSURE(VALUE,UNICODE,UNITS)
134. RETURN
135. C
136. 'DENSITY' TABLE.
137. C
138. 17 CALL DENSITY(VALUE,UNICODE,UNITS)
139. RETURN
140. C
141. 'DEEBEE' TABLE.
142. C
143. 18 CALL DEEBEE(VALUE,UNICODE,UNITS)
144. RETURN
145. 102 FORMAT(' ERROR--VARIABLE NAME CODE',I3,' OUT OF RANGE.')
146. END

```

9FOR,54

M,CVTPRO,CVTPRO

```

1. SUBROUTINE CVTPRO
2. REAL VAL,VALUE
3. IMPLICIT INTEGER(A-Z)
4. C
5. C PROGRAM TO CONVERT FROM A VALUE IN SPECIFIED UNITS
6. C TO THE APPROPRIATE VALUE IN REQUESTED UNITS.
7. C
8. C *TABLE* GIVES DIRECTION TO APPROPRIATE TABLES.
9. C
10. DIMENSION TABLE(99),UNITS(4),UNIT(2)
11. DATA (TABLE(I),I=1,99)/0,1,1,99/0,1,1,1,4,2,4,3,5,4,2,5,7,6,
12. 2,7,2,8,3,19,3,9,3,20,9,10,
13. 17,19,2,14,3,13,2,18,4,19,2,17,
14. 2,19,3,15,3,12,3,16,19,0,8,1,19,
15. 2,13,16,18,4,16,19,2,7,4/
16. DATA ANSWER/'YES' */
17. C
18. C ASK FOR INPUT OF NUMBER OF CONVERSIONS TO BE MADE.
19. C
20. 201 PRINT 101
21. READ(5,102) NCON
22. IF(NCON.LE.0) RETURN
23. NCON=NCON
24. DO 209 I=1,NCON
25. C
26. C ASK FOR INPUT OF VALUE, UNITS FROM, UNITS TO,
27. C
28. 00 203 J=1,J
29. PRINT 103
30. READ(5,102) VAL,UFRM,UTO
31. C
32. C ARE UNITS FROM AND UNITS TO LEGITIMATE?
33. C
34. IF(UFRM.GE.1.AND.UFRM.LE.99) GO TO 202
35. C
36. C ELSE, PRINT ERROR.
37. C
38. PRINT 104,UFRM
39. GO TO 203
40. 202 IF(UTO.GE.1.AND.UTO.LE.99) GO TO 204
41. C
42. C ELSE, PRINT ERROR.
43. C
44. PRINT 105,UTO
45. 203 CONTINUE
46. C
47. C PRINT 'FIVE ERRORS IN A ROW' MESSAGE.
48. C
49. NCON=NCON-1
50. PRINT 106,NCON
51. GO TO 209
52. C
53. C SEE IF BOTH CODES ARE OF SAME TYPE.
54. C

```

```

55.      2J4 IF (TABLE(UFRM).EQ.TABLE(UTO)) GO TO 205
56.      C
57.      C ELSE, PRINT TYPE ERROR.
58.      C
59.      NCON=CON-1
60.      PRINT 107,UFRM,UTO,NCON
61.      GO TO 209
62.      C
63.      C CONVERT TO INTERNAL UNITS.
64.      C
65.      2J5 INDEX=TABLE(UFRM)
66.      UNICDE=UFRM
67.      VALUE=VAL
68.      K=0
69.      206 GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20),INDEX
70.      C
71.      C BRANCH TO CORRECT UNIT-CODE TABLE SUBROUTINE.
72.      C
73.      C
74.      C 'LINEAR' TABLE.
75.      C
76.      1 CALL LINEAR(VALUE,UNICDE,UNITS)
77.      GO TO 207
78.      C
79.      C 'PLANAR' TABLE.
80.      C
81.      2 CALL PLANAR(VALUE,UNICDE,UNITS)
82.      GO TO 207
83.      C
84.      C 'VOLUME' TABLE.
85.      C
86.      3 CALL VOLUME(VALUE,UNICDE,UNITS)
87.      GO TO 207
88.      C
89.      C 'SHORT TIME' TABLE.
90.      C
91.      4 CALL SHRTME(VALUE,UNICDE,UNITS)
92.      GO TO 207
93.      C
94.      C 'LONG TIME' TABLE.
95.      C
96.      5 CALL LGTME(VALUE,UNICDE,UNITS)
97.      GO TO 207
98.      C
99.      C 'VELOCITY' TABLE.
100.      C
101.      6 CALL SPEEDS(VALUE,UNICDE,UNITS)
102.      GO TO 207
103.      C
104.      C 'TEMPERATURE' TABLE.
105.      C
106.      7 CALL TMPTUR(VALUE,UNICDE,UNITS)
107.      GO TO 207
108.      C
109.      C 'ANGLE' TABLE.
110.      C
111.      8 CALL ANGLES(VALUE,UNICDE,UNITS)

```


112.		GO TO 207
113.	C	
114.	C	'PER CENT' TABLE.
115.	C	
116.		9 CALL PERCENTIVALUE,UNICODE,UNITS)
117.		GO TO 207
118.	C	
119.	C	'PRESSURE' TABLE.
120.	C	
121.		10 CALL PRESSUREVALUE,UNICODE,UNITS)
122.		GO TO 207
123.	C	
124.	C	'ACOUSTIC PRESSURE' TABLE.
125.	C	
126.		11 CALL ACPRESSIVALUE,UNICODE,UNITS)
127.		GO TO 207
128.	C	
129.	C	'FREQUENCY' TABLE.
130.	C	
131.		12 CALL CYCLESIVALUE,UNICODE,UNITS)
132.		GO TO 207
133.	C	
134.	C	'ENERGY' TABLE.
135.	C	
136.		13 CALL ENERGYVALUE,UNICODE,UNITS)
137.		GO TO 207
138.	C	
139.	C	'POWER' TABLE.
140.	C	
141.		14 CALL POWERVALUE,UNICODE,UNITS)
142.		GO TO 207
143.	C	
144.	C	'INTENSITY' TABLE.
145.	C	
146.		15 CALL INTENSIVALUE,UNICODE,UNITS)
147.		GO TO 207
148.	C	
149.	C	'DB LOSS/ATTENUATION' TABLE.
150.	C	
151.		16 CALL DBLOSSIVALUE,UNICODE,UNITS)
152.		GO TO 207
153.	C	
154.	C	'PRESSURE LEVEL' TABLE.
155.	C	
156.		17 CALL PRESSLEVELVALUE,UNICODE,UNITS)
157.		GO TO 207
158.	C	
159.	C	'SOURCE LEVEL/TARGET STRENGTH' TABLE.
160.	C	
161.		18 CALL SOURCELEVELVALUE,UNICODE,UNITS)
162.		GO TO 207
163.	C	
164.	C	'MISCELLANEOUS' TABLE.
165.	C	
166.		19 PRINT III,UFPM,UTO
167.		GO TO 209
168.	C	

```

169. C 'DENSITY' TABLE.
170. C
171. 23 CALL DENSITY(VALUE,UNICODE,UNITST
172. GO TO 207
173. 237 IF(K.EQ.1) GO TO 208
174. C
175. C SAVE 'FROM' UNITS.
176. C
177. UNIT(1)=UNITS(1)
178. UNIT(2)=UNITS(2)
179. C
180. C CONVERT FROM INTERNAL UNITS.
181. C
182. UNICODE=-UTO
183. K=1
184. GO TO 204
185. C
186. C WRITE CONVERSION.
187. C
188. 208 PRINT JOB,VAL,UNIT(1),UNIT(2),VALUE,UNITS(1),UNITS(2)
189. NCON=NCON+1
190. 209 CONTINUE
191. C
192. C DO AGAIN?
193. C
194. PRINT 199
195. READ(5,110) RESPNS
196. IF (RESPNS.EQ.ANSWER) GO TO 201
197. RETURN
198. 101 FORMAT(' ENTER NUMBER OF CONVERSIONS TO BE MADE. ')
199. 102 FORMAT(' ')
200. 103 FORMAT(' ENTER VALUE, "UNITS FROM", AND "UNITS TO". ')
201. 104 FORMAT(' ERROR--UNITS FROM= CODE',13,' NOT LEGIT...TRY AGAIN. ')
202. 105 FORMAT(' ERROR--UNITS TO= CODE',13,' NOT LEGIT...TRY AGAIN. ')
203. 106 FORMAT(' HOLD IT! THAT IS THREE ERRORS IN A ROW! PLEASE CHECK CODE
204. 85 CAREFULLY. YOU HAVE',13,' CONVERSIONS LEFT. ')
205. 107 FORMAT(' ERROR--UNITS FROM= CODE',13,' NOT OF SAME TYPE AS "UNIT
206. 85 TO= CODE',13,' .',/, ' YOU HAVE ',13,' CONVERSIONS LEFT. ')
207. 108 FORMAT('G12.5,1H ,2A6.1 = ',G12.5,1H ,2A6)
208. 109 FORMAT(' WOULD YOU LIKE TO DO MORE? ')
209. 110 FORMAT(A6)
210. 111 FORMAT(' NOTA BENE--"UNITS TO" CODE ',13,' NOT. /
211. * CONVERTIBLE TO "UNITS FROM" CODE ',13,/)
212. END

```

RTOR,SA N,CYCLES,CYCLES

```

1. SUBROUTINE CYCLES(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF CYCLES MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(3)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,3)/77,78,79/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,J),J=1,3),J=1,3)/'HERTZ','',
18. C 'KILOGH','RTZ','C','P','M','',
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(3)='HERTZ'
23. C UNITS(4)=' '
24. C
25. C IF UNCODE=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF UNCODE.EQ.0) UNCODE=-77
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA (FACTOR(1,1),1=1,3)/1.000,1.003,6.001/
32. C
33. C IF CODE IS IN TABLE, PERFORM CONVERSION, ALSO, ENTER UNITS ALPHA
34. C CODE INTO 'UNITS'.
35. C
36. C ICODE=ABS(UNICODE)
37. C DO 2 I=1,3
38. C IF (ICODE.EQ.ENTRY(1,I)) GO TO 2
39. C IF UNCODE.GE.0) VALUE=VALUE*FACTOR(I)
40. C IF UNCODE.LT.0) VALUE=VALUE/FACTOR(I)
41. C DO 1 J=1,2
42. C UNITS(J)=ENTRY(J,1)
43. C 1 CONTINUE
44. C
45. C AT THIS POINT, CONVERSION IS COMPLETE.
46. C
47. C RETURN
48. C
49. C ELSE, CHECK REST OF TABLE.
50. C
51. C 2 CONTINUE
52. C
53. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
54. C

```

55. UNITS(1)=UNITS *
56. UNITS(2)=ERROR *
57. PRINT 101,1CODE
58. 161 FORMAT(' ERROR--UNIT CODE',13,' NOT IN FREQUENCY TABLE.')

59. RETURN
60. END

GELT,L N,DATECK,DATELCK

```

1. SUBROUTINE DATECK (IDATE,FODATE,ITIM,FTIM,*)
2. IMPLICIT INTEGER (A-Z)
3.
4. C INCLUDE RETREV.LIST
5. C
6. C
7. C
8. C
9. C
10. C
11. C
12. C
13. C
14. C
15. C
16. C
17. C
18. C
19. C
20. C
21. C
22. C
23. C
24. C
25. C
26. C
27. C
28. C
29. C
30. C
31. C
32. C
33. C
34. C
35. C
36. C
37. C
38. C
39. C
40. C
41. C
42. C
43. C
44. C

```

IYR = IDATE / 10000
 FYR = FDATE / 10000
 IMO = (IDATE - IYR * 10000) / 100
 FMO = (FDATE - FYR * 10000) / 100
 IDAY = (IDATE - IYR * 10000 - IMO * 100)
 FDAY = (FDATE - FYR * 10000 - FMO * 100)
 TMP = FYR * 12 - (12 - FMO)
 TMP1 = IYR * 12 - (12 - IMO)
 IF (TMP - TMP1 .GE. 12) GO TO 50
 IF (IMON .EQ. 0 .OR. IMON .EQ. 12) GO TO 50
 GO TO 101,IMON
 IF (FMO .GT. IMO) GO TO 20
 IF (MONTHS11) .GE. IMO .OR. MONTHS11) .LE. FMO) GO TO 50
 GO TO 30
 20 IF (MONTHS11) .GE. IMO .AND. MONTHS11) .LE. FMO) GO TO 50
 30 CONTINUE
 RETURN 5
 50 IF (FYR .EQ. 0) GO TO 40
 IF (FYR .LT. FYR1) RETURN 5
 40 IF (TOYR .EQ. 0) GO TO 70
 IF (IYR .GT. IYR1) RETURN 5
 70 IF (FDAY .EQ. 0) GO TO 80
 IF (FDAY .LT. FDAY1) RETURN 5
 80 IF (IDAY .EQ. 0) GO TO 100
 IF (IDAY .GT. IDAY1) RETURN 5
 100 IF (FTIM .EQ. 0 .AND. TOTIM .EQ. 0) RETURN
 IF (TOTIM .LT. FTIM) GO TO 110
 IF (FTIM .LT. FTIM1 .OR. ITIM .GT. TOTIM) RETURN 5
 RETURN
 110 IF (FTIM .GE. FTIM1 .AND. ITIM .LE. TOTIM) RETURN 5
 RETURN
 END

SUBROUTINE DATLST (OPTION,CHAR,NUNTP)

A-31

```

55. GO TO 8
56. C
57. C ERROR
58. C
59. C WRITE (6,7)
60. 7 FORMAT (1, '... FORMAT ERROR OR INVALID DATA TYPE ...')
61. GO TO 1
62. C
63. C FIND WHICH RUNS ARE TO BE PRINTED
64. C
65. 8 WRITE (6,9)
66. 9 FORMAT (1, 'ENTER EXPER. NO., CRUISE NO., STATION NO., RUN NO., ENTER ...')
67. 10 CALL 'ENTER' (END, 1)
68. 11 READ (5, 11) (END=1, ERR=17) (CHAR(1), 1=1, 80)
69. 12 FORMAT (10A1)
70. 13 WRITE (6, 11) (CHAR(1), 1=1, 80)
71. 14 FORMAT (1X, 80A1)
72. 15 CRU1 = -99999999
73. 16 STAI = -99999999
74. 17 RUN1 = -99999999
75. 18 CALL FREEFO (CHAR, WORD, EXPT, NWORDS, 9, 817)
76. 19 W = . . .
77. 20 FLD (10, 6, 8) = FLD (10, 6, EXPT)
78. 21 IF (W.EQ.'E') GO TO 900
79. 22 IF (W.NE.'A'.AND.NWORDS.GE.1) GO TO 15
80. 23 CALL GETINT
81. 24 USERAP = .FALSE.
82. 25 ALL = .TRUE.
83. 26 GO TO 10
84. C
85. C CHECK FOR ERROR
86. C
87. 15 DO 16 I = 1, NWORDS
88. 16 IF (ECR(I).LT.1.OR.ECSR(I).GT.ERRCK(I)) GO TO 17
89. 17 GO TO 19
90. 18 WRITE (6, 18)
91. 19 FORMAT (10, '... FORMAT ERROR OR NUMBER OUT OF RANGE ...')
92. 20 GO TO 8
93. C
94. 19 CALL GETINT
95. 20 USERAP = .FALSE.
96. 21 ALL = .FALSE.
97. 22000 CALL GETRUN (IDATYPE, EXPNO, CRUNO, STANO, 822)
98. 23 GO TO 24
99. 24 IF (NRUNSF.GT.0) GO TO 8
100. 25 WRITE (6, 23)
101. 26 FORMAT (10, '... SPECIFIED RECORD(S) NOT FOUND ...')
102. 27 GO TO 8
103. C
104. 10 CALL GETRUN (IDATYPE, EXPNO, CRUNO, STANO, 8900)
105. C
106. C PRINT HEADINGS
107. C
108. 24 PRINT 11
109. 11 FORMAT ('1')
110. 25 IF (IDATYPE.EQ. 4) GO TO 20
111. C

```



```

112. WRITE (6,12)
113. 12 FORMAT (////, 'ACOUSTIC DATA')
114. 60 TO 25
115. 20 WRITE (6,21)
116. 21 FORMAT (////, 'ENVIRONMENTAL DATA')
117. 5 GET EXPERIMENT NAME AND PRINT
118. 25 CALL NAMEIT(1,EXPNO,NAME,NR)
119. WRITE (6,24) EXPNO,NAME(1),NR
120. 24 FORMAT (5,'EXPERIMENT: ',16,19,32,64)
121. WRITE (6,27) CRUNO,STANO,MUMNO
122. 27 FORMAT (5,'CRUISES: ',16,19,75,'STATION: ',16,19,
123. 5 15,'RUN: ',16,19)
124. C
125. C SET UP AREA & DATE FOR PRINT
126. C
127. C RINS = 'N'
128. IF (RILAT .GE. 0) 60 TO 30
129. RINS = 'S'
130. RILAT = -RILAT
131. C
132. 30 RFNS = 'N'
133. IF (RFLAT .GE. 0) 60 TO 35
134. RFNS = 'S'
135. RFLAT = -RFLAT
136. C
137. 35 RIER = 'E'
138. IF (RILON .GE. 0) 60 TO 40
139. RIER = 'W'
140. RILON = -RILON
141. C
142. 40 RFEW = 'E'
143. IF (RFLON .GE. 0) 60 TO 45
144. RFEW = 'W'
145. RFLON = -RFLON
146. C
147. 45 RIYR = RIDAY / 10000
148. RFYR = RFDAY / 10000
149. RIMO = (RIDAY - RIYR*10000) / 100
150. RFMO = (RFDAY - RFYR*10000) / 100
151. RIDAY = RIDAY - RIYR*10000 - RIMO*100
152. RFDAY = RFDAY - RFYR*10000 - RFMO*100
153. C
154. IF (RITIM .LE. 2400) 60 TO 50
155. RITIM = 9999
156. RIZON = 'O'
157. 50 IF (RITIM .LE. 2400) 60 TO 55
158. RITIM = 9999
159. RIZON = 'O'
160. C
161. C PRINT AREA, DATE & TIME
162. C
163. 55 R11 = RILAT / 1000
164. R12 = RILON / 1000
165. R13 = RFLAT / 1000
166. R14 = RFLON / 1000
167. C
168. R21 = (RILAT - R11*1000) / 10

```



```

226.      IF (DATYPE .EQ. 3) GO TO 80
227.      C
228.      WRITE (6,70) (INFO(I),I=1,9)
229.      70 FORMAT (15,'SHIP'S NAME: ',9A0)
230.      C
231.      IF (FLD132,1,1,STAT121).EQ.0) GO TO 80
232.      WRITE (6,75)
233.      75 FORMAT (15,'BATHMETRY AVAILABLE - SEE NOTES')
234.      C
235.      80 IF (NCON .EQ. 0) GO TO 100
236.      WRITE (6,85)
237.      85 FORMAT (17,'15,000 CONSTANTS ...')
238.      IF (DATYPE.NE.3) GO TO 82
239.      WRITE (6,81)
240.      81 FORMAT (160,'ACOUSTIC')
241.      GO TO 84
242.      82 WRITE (6,83)
243.      83 FORMAT (158,'ENVIRONMENTAL')
244.      84 WRITE (6,86)
245.      86 FORMAT (115,'NAME',I33,'VALUE',I98,'UNITS',I61,'METHOD',/I)
246.      C
247.      DO 95 I=1,NCON
248.      DUM(I) = COM13,1)
249.      CALL CHECK (DUM(I),ICON(1,1),UNITS)
250.      CALL NAMEGT (ICON(1,1),NAME)
251.      IF (ICON(1,1).EQ.0) ICON(4,1) = 1
252.      WRITE (6,87) (NAME(J),J=1,2),COM13,1),UNITS(1),UNITS(2),ICON(4,1)
253.      87 FORMAT (11,2A6,2A6,6A5,4A1,2A6,2A14)
254.      95 CONTINUE
255.      C
256.      100 IF (NDAYS .EQ. 0) GO TO 600
257.      C
258.      C      LOOP TO SET PARAMETER POINTERS TO 1
259.      C
260.      DO 110 I=1,30
261.      PSUB(I) = 1
262.      110 CONTINUE
263.      C
264.      DSN = 1      CURRENT DATA SET # FOR PARAMETERS
265.      RKT = 0      LIMIT FOR ROB COUNTING
266.      WRITE (6,111)
267.      111 FORMAT (11)
268.      DO 500 I=1,NDUSED
269.      C
270.      C      PRINT DATA SET N
271.      C
272.      WRITE (6,112) NRS(I)
273.      112 FORMAT (175,'DATA SET ',I2)
274.      C
275.      IF (NPAR .EQ. 0) GO TO 200      0 GO DO VARIABLES
276.      IF (NRS(I) .EQ. 1) GO TO 120
277.      C
278.      114 J = NPAR
279.      115 PSUB(J) = PSUB(J) + 1
280.      IF (PSUB(J) .LE. 1PAR(1,J)) GO TO 117
281.      PSUB(J) = 1
282.      J = J - 1

```

```

283.      GO TO 115
284.      117 DSN = DSN + 1
285.      IF (DSN.NE.NKS(11)) GO TO 114
286.      C
287.      C      PRINT OUT PARAMETERS
288.      C
289.      120 WRITE (6,123)
290.      123 FORMAT (//,15,'... PARAMETERS ...')
291.      IF (DATE.NE.3) GO TO 125
292.      WRITE (6,124)
293.      124 FORMAT (1760,'ACOUSTIC')
294.      GO TO 127
295.      125 WRITE (6,126)
296.      126 FORMAT (1758,'ENVIRONMENTAL')
297.      127 WRITE (6,86)
298.      C
299.      DO 190 K=1,NPAR
300.      PSB = PSUB(K)+4
301.      DUM(1) = PAR(PSB,K)
302.      CALL CHECK(DUM(1),IPAR(2,K),UNITS)
303.      CALL NAMEGET (IPAR(2,K),NAME)
304.      IF (IPAR(4,K).EQ.0) IPAR(4,K) = 1
305.      WRITE (6,87) (NAME(I),I=1,2),PAR(PSB,K),UNITS(1),
306.      UNITS(2),IPAR(4,K)
307.      190 CONTINUE
308.      C
309.      C      PRINT VARIABLES
310.      C
311.      200 WRITE (6,201)
312.      201 FORMAT (//,15,'... VARIABLES ...')
313.      C
314.      IF (OPTION.EQ.4) CALL SHORT (NKS(11),DATYPE,IVLRNG,UNITS,NAME,3500)
315.      L1 = 1
316.      210 L2 = L1 + 4
317.      IF (L2.GT. NVAR) L2 = NVAR
318.      C
319.      L3 = 0
320.      DO 250 L4 = L1,L2
321.      L3 = L3 + 1
322.      IF (L1.EQ.1) SAVE(L3) = DATA(L4)
323.      CALL NAMEGET (IVAR(1,L4),VBUF(2,(L3-1)))
324.      250 CONTINUE
325.      T1 = 2*L3
326.      WRITE (6,255) (VBUF(ILP),LP=1,T1)
327.      255 FORMAT (//,18,5(1X,2X))
328.      C
329.      L3 = 0
330.      DO 270 L4 = L1,L2
331.      L3 = L3 + 1
332.      DUM(1) = SAVE(L3)
333.      CALL CHECK (DUM(1),IVAR(1,L4),UNITS)
334.      FLD(10,36,VBUF(2*(L3-1))) = UNITS(1)
335.      FLD(10,36,VBUF(2*(L3-1))) = UNITS(2)
336.      270 CONTINUE
337.      T1 = 2 * L3
338.      WRITE (6,275) (VBUF(ILP),LP=1,T1)
339.      275 FORMAT (18,5(1X,2X))

```

```

340. C
341. WRITE (6,290) (IVAR(3,LP),LP=1,L2)
342. 290 FORMAT (14,'METHOD',14,'(19X,14)')
343. WRITE (6,291)
344. 291 FORMAT (1)
345. C
346. C COMPUTE BEGINNING SUBSCRIPT FOR DATA
347. C IN DATA SET NRS(1)
348. C
349. RKT = G
350. IF (NRS(1) .EQ. 1) GO TO 300
351. JK = NRS(1)-1
352. DO 295 I=1,JK
353. MAT = RKT + NROWS(I,K)
354. 295 CONTINUE
355. 300 CONTINUE B NOT END OF A GO LOOP
356. C
357. C LOOPS TO PRINT DATA VALUES
358. C
359. I1 = NRS(1)
360. L7 = NROWS(1)
361. DO 450 L5 = 1,L7
362. RKT = RKT + 1
363. C
364. L3 = 0
365. DO 400 L4 = 1,L3
366. L3 = L3 + 1
367. DSUB = RKT+NVAR - NVAR
368. DUM(I) = DATA(DSUB+L4)
369. CALL CHECK (DUM(I),IVAR(1,L4),UNITS)
370. VBUF(L3) = DATA(DSUB+L4)
371. 400 CONTINUE
372. C
373. WRITE (6,425) L5,(VBUF(LP),LP=1,L3)
374. 425 FORMAT (12,15,5(12,5,1X))
375. C
376. C 450 CONTINUE
377. C
378. IF (L2 .EQ. NVAR) GO TO 490
379. L1 = L2 + 1
380. GO TO 210
381. 490 WRITE (6,111)
382. C
383. C 500 CONTINUE
384. C
385. 600 IF (ALL) GO TO 10
386. GO TO 2000
387. C
388. C
389. 900 WRITE (6,901)
390. 901 FORMAT (17,'17','LISTING OF RUN DATA COMPLETED')
391. RETURN
392. END

```


QFOR,34 N,DBLOSS,,DBLOSS

```
1. SUBROUTINE DBLOSS(VALUE,UNICODE,UNITS)
2. C
3. C TABLE FOR DB LOSS AND ATTENUATION CONVERSION FACTORS.
4. C
5. DIMENSION FACTOR(8)
6. INTEGER ENTRY(3,8),UNITSTR(4),UNICODE
7. C
8. C PUT UNIT CODES INTO ENTRY.
9. C
10. DATA (ENTRY(1,1),1,1,1,8)/87,81,82,90,92,93,94,95/
11. C
12. C ENTER CONVERSION FACTORS.
13. C
14. DATA (FACTOR(1,1),1,1,8)/.8,-59.2,0,-60,0,0,-.8,59.2,60./
15. C
16. C PUT IN ALPHA UNITS.
17. C
18. DATA ((ENTRY(1,1),1,2,3),J(1,8)/.08//1 'YARD '
19. '08//1 'YARD '08//1 'METER '
20. '08//1 'METER',.08/MET',PER '
21. '08/YAR',D '08/KYA',RD '
22. '08/ME',.TER '
23. C
24. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
25. C
26. C UNITS(1)='UNITS '
27. C UNITS(2)='ERROR '
28. C
29. C RETRIEVAL IN STANDARD DB LOSS UNITS.
30. C
31. C IF (UNICODE.NE.0) GO TO 1
32. C UNCODE=-82
33. C
34. C ENTER INTERNAL UNITS.
35. C
36. C UNITS(3)='08//1'
37. C UNITS(4)='METER '
38. C GO TO 2
39. C
40. C RETRIEVAL IN STANDARD ATTENUATION UNITS.
41. C
42. C 1 IF (UNICODE.NE.-1) GO TO 2
43. C UNCODE=-92
44. C
45. C ENTER INTERNAL UNITS.
46. C
47. C UNITS(3)='08/MET'
48. C UNITS(4)='PER '
49. C
50. C IF CODE IS IN TABLE, PERFORM CONVERSION, ALSO, ENTER UNITS ALPHA
51. C CODE INTO 'UNITS'.
52. C
53. C 2 ICODE=1ABS(UNICODE)
54. C DO 3 1=1,0
```

```

55. IF(ICODE.NE.ENTRY(1,1)) GO TO 3
56. IF(UNICD.GT.0) VALUE=VALUE*FACTOR(1)
57. IF(UNICD.LT.0) VALUE=VALUE-FACTOR(1)
58. UNITS(1)=ENTRY(2,1)
59. UNITS(2)=ENTRY(3,1)
60. C
61. C AT THIS POINT, CONVERSION IS COMPLETE.
62. C
63. RETURN
64. 3 CONTINUE
65. C
66. C ELSE, UNIT CODE IS NOT CONVERTIBLE.
67. C
68. PRINT 101,ICODE
69. PRINT 112
70. 112 FORMAT(' ERROR--UNIT CODE NOT CONVERTIBLE(')
71. UNITS(1)=UNITS
72. RETURN
73. 101 FORMAT(' ERROR--UNIT CODE',13,' NOT IN DB LOSS TABLE,')
74. END

```

QFOR,SA N,DEEBEE,DEEBEE

```

1. SUBROUTINE DEEBEE(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF DB UNITS.
4. C
5. C INTEGER ENTRY(3,27),UNITS(4),UNICODE
6. C
7. C PUT CODES INTO 'ENTRY'.
8. C
9. C DATA (ENTRY(1,1),1=1,27)/39,40,41,58,66,67,68,69,72,73,83,80,96,
10. C 87,81,82,90,92,93,94,95,64,65,91,101,102,103/
11. C
12. C ENTER ALPHA UNITS.
13. C
14. C DATA ((ENTRY(1,J),J=1,27)/('DB/YD',1,2, '
15. C //IUBR',12/S/HZ',1,CM3/ARI',9AM '
16. C 'PRESSR',1, 'RATIO',1, 'DB//IM',1, 'ATT '
17. C 'DB//IM',1, 'M+2',1, 'DB//IM',1, 'CM+2',1,
18. C //IENG',1, 'SC/CH2',1, 'DB//IE',1, 'RG/CH2',1,
19. C 'IENG/S',1, 'EC/CM',1, //JLE',1, 'CM2-HZ',1,
20. C 'DB',1, //ERG',1, 'CM2QIY',1,
21. C 'DB',1, //DB//I',1, 'KYARD',1,
22. C 'DB//I',1, 'METER',1, 'DB//I',1, 'KMETER',1,
23. C 'DB/MET',1, 'ER',1, 'DB/YAR',1, 'D',1,
24. C 'DB/XAY',1, 'RD',1, 'DB/KMET',1, 'TER',1,
25. C 'DB//IU',1, 'BARGIM',1, 'DB//IU',1, 'BARDIY',1,
26. C 'DB//IU',1, 'PAPIM',1, 'DB//IU',1, 'PADIYD',1,
27. C //IUBA',1, 'R2SOIY',1, //IUPA',1, '2 SWIM',1/
28. C
29. C IF UNICODE.EQ.0, THEN SET UNITS TO "DB AS MEAS.".
30. C
31. C UNITS(1)="DB AS "
32. C UNITS(2)="MEAS."
33. C
34. C SET INDICATOR FOR CASE UNPACK IS BYPASSED.
35. C
36. C ITST=0
37. C
38. C IF RETRIEVING, UNPACK VALUE AND UNIT CODE.
39. C
40. C IF(UNICODE.EQ.1) GO TO 1
41. C ICODE=VALUE
42. C IF(UNICODE.EQ.0.AND.ICODE.EQ.-99999999) RETURN
43. C IF(UNICODE.NE.0.AND.ICODE.EQ.-99999999) GOTO 5
44. C UNICODE=0
45. C UNICODE=ABS(FLD(29,7,VALUE))
46. C
47. C IF BE AREN'T RETRIEVING, THEN RETURN.
48. C
49. C IF(ICODE.EQ.0.AND.UNICODE.EQ.0) RETURN
50. C
51. C SET INDICATOR FOR CASE PACK IS BYPASSED.
52. C
53. C 5 )TST=1
54. C

```

55. C TEST TO SEE IF CODE IS IN TABLE.

56. C

57. 1 ICODE=IABS(UNICODE)

58. 00 2 1-1,27

59. IF(ICODE.EQ.ENTRY(1,1)) GO TO 3

60. 2 CONTINUE

61. C

62. C IF CODE NOT FOUND, WRITE ERROR:

63. C

64. PRINT 101,ICODE

65. PRINT 102,UNICODE,VALUE,ITST

66. 102 FORMAT(' UNICOD=115,' VALUE(OCTAL)=',1012,' ITST=',115)

67. UNITS(1) = 'UNITS'

68. UNITS(2) = 'ERROR'

69. RETURN

70. C

71. C PACK CODE AND VALUE, IF NECESSARY.

72. C

73. 3 IF(ITST.EQ.1) GO TO 4

74. FLD(29,7,VALUE)=FLD(29,7,UNICODE)

75. C

76. C SET UNITS, RETURN:

77. C

78. 4 UNITS(1)=ENTRY(2,1)

79. UNITS(2)=ENTRY(3,1)

80. UNITS(3)=ENTRY(4,1)

81. UNITS(4)=ENTRY(5,1)

82. RETURN

83. 101 FORMAT(' ERROR--UNIT CODE',13,' NOT IN DEEBEE TABLE.')

84. END

QFOR,SA N,DENSTY,,DENSTY

```

1. SUBROUTINE DENSTY(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF DENSITY MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(11)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,3)/45,46,47/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,1),1=2,3),J=1,3)/'GMS/CM','GMS/CM','GMS/CM',
18. C 'KG/MT','GMS/CM',
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(3)='GMS/CM'
23. C UNITS(4)='GMS/CM'
24. C
25. C IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF (UNICDE.EQ.0) UNICDE=-90
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA (FACTOR(1,1),1=1,3)/1.000,1.003,7.74922304/
32. C
33. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
34. C UNITS(1)='UNITS'
35. C UNITS(2)='ERROR'
36. C
37. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
38. C CODE INTO 'UNITS'.
39. C
40. C ICODE=ABS(UNICDE)
41. C DO 2 1=1,3
42. C IF (ICODE.NE.ENTRY(1,1)) GO TO 2
43. C IF (UNICDE.GE.0) VALUE=VALUE*FACTOR(11)
44. C IF (UNICDE.LT.0) VALUE=VALUE/FACTOR(11)
45. C UNITS(1)=ENTRY(2,1)
46. C UNITS(2)=ENTRY(3,1)
47. C
48. C AT THIS POINT, CONVERSION IS COMPLETE.
49. C
50. C RETURN
51. C
52. C ELSE, CHECK REST OF TABLE.
53. C
54. C 2 CONTINUE

```

```

55. C      IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
56. C
57. C
58.      PRINT I2I,ISCODE
59.      I2I FORMAT(' ERROR--UNIT CODE',I3,' NOT IN DENSITY TABLE.')
```

```

      RETURN
      END
```

WFOH,SA N-DESCB,DESCB

```
1: SUBROUTINE DESCB (ITEMS, NITEMS, ISET)
2:
3: C PROGRAM TO PRINT THE NUMBER OF RUNS (IF NONZERO) OF THE ACOUSTIC
4: C OR ENVIRONMENTAL DESCRIPTIONS (ISET=0 FOR ACOUSTIC, ISET=7 FOR
5: C ENVIRONMENTAL).
6:
7: DIMENSION ITEMS(1), NAME(10)
8: WRITE (6,10)
9: 10 FORMAT (143, 'NUMBER', ' CODE', 112, 'DESCRIPTOR NAME', 143, 'OF RUNS',
10: 17)
11: DO 30 I = 1, NITEMS
12: IF (ITEMS(I).EQ.0) GO TO 30
13: CALL NAMEIT (ISET, I, NAME, NR)
14: IF (NR.GT.5) NR = 5
15: WRITE (6,20) ITEMS(I), I, NAME(I), J=1, NR)
16: 20 FORMAT (142, 16, 14, 13, ' - ', 5A6)
17: 30 CONTINUE
18: RETURN
19: END
```

QELT,L N.DSKRED,DSKRED

```
1. SUBROUTINE DSKRED(IN,IS,ADR)
2. C
3. C
4. CALL SETADR(10,IN)
5. IN = IS * 20
6. CALL NTRAM(10,2,IN,ADR,L)
7. CALL NTRAM(10,22)
8. C
9. IF (L .GT. 0) RETURN
10. C
11. WRITE (6,10) IN,IS,L
12. 10 FORMAT('ERROR IN DSKRED',J110)
13. RETURN 0
14. END
```


BELT L N-EAST-EAST

```
1.      INTEGER FUNCTION EAST (A,B)
2.      C
3.      C
4.      C
5.      C
6.      IF (IABS(A-B) .GT. 16000) GO TO 10
7.      EAST = MAX(A,B)
8.      RETURN
9.      C
10.     10 EAST = 0
11.     IF (A .LE. 0) EAST = A
12.     RETURN
13.     END
```

OFOR,50 N,ENERGY,ENERGY

```
1. SUBROUTINE ENERGY(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF ENERGY MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,5),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(5)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1,1,5), (1,2,3), (1,3,1,5), (1,4,5), (1,5,1,5)
14. C DATA (ENTRY(1,1),1,1,5), (1,2,3), (1,3,1,5), (1,4,5), (1,5,1,5)
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA (ENTRY(1,1),1,1,5), (1,2,3), (1,3,1,5), (1,4,5), (1,5,1,5)
18. C
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C
23. C UNITS(3)=KGM ME
24. C UNITS(4)=TENS
25. C
26. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
27. C
28. C IF (UNICODE.EQ.0) UNICODE=00
29. C
30. C SET CONVERSION FACTORS.
31. C
32. C DATA (FACTOR(1,1),1,1,5), (1,2,3), (1,3,1,5), (1,4,5), (1,5,1,5)
33. C
34. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
35. C
36. C UNITS(1)=UNITS
37. C UNITS(2)=ERROR
38. C
39. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
40. C CODE INTO 'UNITS'.
41. C
42. C (CODE=ABS(UNICODE)
43. C DO 2 I=1,5
44. C IF (UNICODE.EQ.ENTRY(1,1)) GO TO 2
45. C IF (UNICODE.EQ.0) VALUE=VALUE*FACTOR(1)
46. C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(1)
47. C UNITS(1)=ENTRY(2,1)
48. C UNITS(2)=ENTRY(3,1)
49. C
50. C AT THIS POINT, CONVERSION IS COMPLETE.
51. C
52. C RETURN
53. C
54. C ELSE, CHECK REST OF TABLE.
```

```

55. C
56. 2 CONTINUE
57. C
58. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
59. C
60. PRINT IQL,ICODE
61. I01 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ENERGY TABLE.')
62. RETURN
63. END

```

GELT,L N,FREEFD,,FREEFD

```

1. SUBROUTINE FNEED (CHAR,WORD,IAORD,NWORDS,MAXIS)
2.
3. C THIS PROGRAM ACCEPTS A STRING OF 80 CHARACTERS CONTAINING A VARIABLE NUMBER
4. C OF DATA VALUES INPUTTED IN FREE-FIELD FORMAT. THE VALUES CAN BE REAL,
5. C INTEGER, OR ALPHANUMERIC. AN ALPHANUMERIC VALUE IS A VARIABLE LENGTH
6. C STRING OF ALPHANUMERIC CHARACTERS BEGINNING WITH AN ALPHABETIC CHARACTER.
7. C MAX
8.
9. IMPLICIT INTEGER (A-Z)
10. REAL WORD
11. DIMENSION CHAR(1),WORD(1),IWORD(1),VALUE(3)
12. DATA COMMA /',',/, Z/0370505050505/
13. NWORDS = 0
14. CURCHR = 1
15. BIT = 0
16. M = 1
17. B = 0
18. CHAR(81) = ' '
19.
20. I = 0
21. I = I + 1
22. IF (I.GT.81) RETURN
23. IF (CHAR(I).EQ.' ') GO TO 1
24. IF (CHAR(I).NE.' ') GO TO 7
25. IF (CURCHR.EQ.1) GO TO 3
26. I NWORDS = NWORDS + 1
27. C FLAG VALUE OF -99999999 STORED IN CASE A BLANK VALUE IS FOUND.
28. IORD(NWORDS) = -99999999.
29. C SKIP DECODE IF BLANK VALUE.
30. IF (I.GT.CURCHR) DECODE (I0,2,VALUE,ERR=5) WORD(NWORDS)
31. FORMAT (1)
32. IWORD(NWORDS) = WORD(NWORDS)
33. C CHECK IF MAX VALUES HAVE BEEN DECODED
34. IF (NWORDS.GE.MAX) GO TO 15
35. BIT = 0
36. M = 1
37. B = 0
38. CURCHR = 1 + 1
39. GO TO 4
40.
41. C CHECK IF ALPHABETIC AND IF FIRST CHARACTER OF STRING.
42. IF (CHAR(1).GT.2.OR.BIT.GT.0) GO TO 8
43. C LOOP TO STORE ALPHANUMERIC STRING
44. NWORDS = NWORDS + 1
45. IORD(NWORDS) = ' '
46. IF (BIT.GE.36) GO TO 11
47. FLD(BIT,6,NORD(NWORDS)) = FLD(I0,6,CHAR(1))
48. BIT = BIT + 6
49. I = I + 1
50. IF (CHAR(1).NE.' ' .AND. CHAR(1).NE.' ') GO TO 9
51. FLD(0,36,IWORD(NWORDS)) = FLD(I0,36,IWORD(NWORDS))
52. GO TO 12
53. C PACK A DATA CHARACTER INTO VALUE.
54. IF (W.GT.3) GO TO 5

```



```

55. FLD(8.6,VALUE(W)) = FLD(10.6,CHAR(I))
56. BIT = BIT + 6
57. I = (BIT/36) + 1
58. B = MOD(BIT,36)
59. FLD(18.6,VALUE(W)) = FLD(10.6,COMMA)
60. GO TO 4
61.
62. C RETURN TO ERROR STATEMENT IN CALLING PROGRAM.
63. S RETURN 6
64.
65. C CHECK IF MORE THAN MAX VALUES HAVE BEEN INPUTTED
66.
67. I = I + 1
68. IF (I.GT.80) RETURN
69. DO 20 J = 1,80
70. 20 IF (CHAR(J).NE.' ') GO TO 25
71. RETURN
72.
73. C TOO MANY VALUES INPUTTED
74.
75. 25 WRITE (6,30) MAX
76. 30 FORMAT (1.000 MORE THAN MAXIMUM OF 13,1 VALUES INPUTTED, EXCESS
77. 1 VALUES IGNORED 0000)
78. RETURN
79.
80. END

```

BFOR,54 N.GETINT,,GETINT

```

1. SUBROUTINE GETINT
2. IMPLICIT INTEGER (A-Z)
3. DIMENSION I8FR(300),DESC(54)
4. C
5. C
6. INCLUDE RUN.LIST
7. C
8. MEAL WVDIR,WAVHT,WAVPRD,SEASTA,WMDIR,WNDVEL,SBLDIR,
9. SBLHT,SBLPRD,HEATMR,BOTDPH,BTHSLP
10. C
11. C
12. INCLUDE CPV.LIST
13. DIMENSION ICOM(4,30),IPAR(30,30),IVLRNG(2,30,50),IEQU(3000)
14. EQUIVALENCE (ICOM(1,1),ICON(1,1)),(IPAR(1,1),IPAR(1,1)),
15. (IVLRNG(1,1),IVLRNG(1,1)),IEQU(1,1))
16. REAL CON,PAR,DATA,VALRNG
17. INCLUDE GETLIST.LIST
18. C
19. C
20. INCLUDE RETNEV.LIST
21. C
22. INCLUDE DATCOM.LIST
23. C
24. USERAP = .TRUE.
25. ALL = .TRUE.
26. C
27. C
28. CALL NTRAN(10,22)
29. CALL NTRAN(11,10)
30. C
31. C READ OR BYPASS RETRIEVE COMMON.
32. C
33. IF (USERAP) GO TO 5
34. CALL SETADR (11,16)
35. GO TO 10
36. C
37. S CALL NTRAN(11,2,120,NLAT,LSTAT)
38. C
39. C NO PARALLEL PROCESSING.
40. C
41. C CALL NTRAN(11,22)
42. C
43. C WRITE 'NTRAN ERROR' IF ANY.
44. C
45. C IF (LSTAT.GT.0) GO TO 10
46. PRINT 101,LSTAT
47. 101 FORMAT(' ERROR--GETINT NTRAN STATUS ',I2)
48. STOP
49. C
50. 10 ICTR=0
51. LASTDA=0
52. NRUNSF = 0
53. C
54. C INITIALIZATION COMPLETE.

```

```

55. C RETURN
56. C
57. C
58. C
59. C
60. C
61. C
62. C
63. C ROUTINE FOR PROCESSING EACH RUN.
64. C
65. C ENTRY GETRUN(DATY,EXPNO,CRUNO,STANO,*)
66. C
67. C WHEN RUN COUNTER IS .GE. NUMBER OF POINTERS, WE ARE DONE.
68. C
69. C IF(ICTR,GE,NPTRSI) RETURN 5
70. C
71. C ICTR=ICTR+1
72. C
73. C CALL NTRAN(1,2,56,DESC,LSTAT)
74. C
75. C NO PARALLEL PROCESSING.
76. C
77. C CALL NTRAN(1,22)
78. C
79. C WRITE NTRAN ERROR, IF ANY.
80. C
81. C IF(LSTAT.GT.0) GO TO 2
82. C PRINT 102,LSTAT,ICTR
83. C 102 FORMAT(' ERROR--GETRUN NTRAN STATUS ',12,' ICTR = ',14,'.')
84. C STOP
85. C
86. C CHECK EXP, CRU, AND STA NUMBER
87. C
88. C 2 EXPNO = DESC(1)
89. C CRUNO = DESC(2)
90. C STANO = DESC(3)
91. C IF (ALL) GO TO 3
92. C IF (EXPNO.NE.EXP1) GO TO 1
93. C IF (CRU1.EQ.-99999999) GO TO 3
94. C IF (CRUNO.NE.CRU1) GO TO 1
95. C IF (STAN1.EQ.-99999999) GO TO 3
96. C IF (STANO.NE.STA1) GO TO 1
97. C
98. C 3 NEXTDA = FLD( 0,18,DESC(4))
99. C LENGTH = FLD(18,18,DESC(4))
100. C
101. C SET ISUB = SUBSCRIPT OF RUN
102. C
103. C ISUB = DESC(5)
104. C
105. C IF (LASTDA .EQ. NEXTDAY) GO TO 15
106. C
107. C CALL USKRED (NEXTDA,LENGTH,IBFH)
108. C
109. C UPDATE LAST DISK ADDRESS PTN
110. C
111. C LASTDA = NEXTDA

```

```

112. C UNPACK DATA TYPE
113. C
114. C
115. C DATATYPE = FLD(10,18,18FR(1))
116. C
117. C IF NOT ACOUSTIC ON ENV, DATA, WRITE ERROR
118. C
119. C IF (DATATYPE.EQ.3 .OR. DATATYPE.EQ.4) GO TO 15
120. PRINT 103, DATATYPE
121. 103 FORMAT (' ERROR -- GETMUN DATA TYPE',12,' ILLEGAL.')
122. STOP
123. C
124. C CHECK RUN NUMBER
125. C
126. 15 RUNNO = FLD(10,18,18FR(1508))
127. IF (USEMAP) GO TO 20
128. IF (ALL) GO TO 18
129. IF (RUN1.EQ.-99999999) GO TO 18
130. IF (RUNNO.NE.RUN1) GO TO 1
131. 18 IF (AETYPE.EQ.3) GO TO 20
132. IF (AETYPE.EQ.2) GO TO 19
133. IF (DATATYPE.EQ.4) GO TO 1
134. GO TO 20
135. 19 IF (DATATYPE.EQ.3) GO TO 1
136. C
137. C FIND PTR TO DATA
138. C
139. 20 DPTR = 18FR(1508 + 9)
140. C
141. C SECAD = FLD( 0,18,DPTR)
142. SLNGTH = FLD(18,18,DPTR)
143. C
144. C CALL AUNPCK (18FR,1508)
145. C
146. C IF (DATATYPE.EQ.4) GO TO 30
147. C
148. C CALL DSKRED (SECAD,SLNGTH,DATA(0)) @ WATCH FOR ZERO SUBSCRIPT
149. GO TO 40
150. C
151. C
152. 30 CALL DSKRED (SECAD,SLNGTH,18FR(2991))
153. DO 35 I=1,9
154. INFO(I) = 18FR(2991+I)
155. 35 CONTINUE
156. C
157. 40 NUSED = DESC(6)
158. DO 50 I=1,50
159. NRS(I) = DESC(6+I)
160. 50 CONTINUE
161. C
162. C CALL TO PERFORM UNIT CONVERSIONS
163. C
164. C CALL UNTCVT
165. C
166. C INCREMENT NUMBER OF RUNS FOUND
167. C
168. RUNSF = RUNSF + 1

```


169.
170.
171.
172.
173.

C
C
C

DONE

RETURN
END

BELT,L N,IDNAME,,IDNAME

```
1. SUBROUTINE IDNAME (CHAR,ID,NAME,FLAG)
2.
3. C THIS ROUTINE DECODES THE USER NAME AND ID
4.
5. IMPLICIT INTEGER (A-Z)
6. DIMENSION CHAR(1),NAME(1)
7. DATA A/O082605050505/ 2/0370505050505/
8. C
9. C DECODE NAME
10. C
11. FLAG = 0
12. WORD = 1
13. BIT = 0
14. DO 15 I = 1,5
15. 15 NAME(I) = ' '
16. I = 0
17. 40 I = 1
18. IF (I.GT.80) GO TO 25
19. IF (CHAR(I).EQ.' ') GO TO 40
20. IF (CHAR(I).LT.A.OR.CHAR(I).GT.Z) GO TO 25
21. DO 20 J = 1,30
22. FLD (BIT,6,NAME(WORD)) = FLD (0,6,CHAR(I))
23. BIT = BIT + 6
24. I = I + 1
25. IF (CHAR(I).EQ.' ') GO TO 5
26. IF (BIT.LT.36) GO TO 50
27. BIT = 0
28. WORD = WORD + 1
29. CONTINUE
30. 50 GO TO 5
31. C
32. C ERROR
33. C
34. 25 WRITE (6,30)
35. 30 FORMAT ('... ERROR ENTERING USER NAME OR ID ...')
36. FLAG = 1
37. RETURN
38. C
39. C DECODE ID
40. C
41. 5 BIT = 0
42. 10 I = 1
43. IF (I.GT.80) GO TO 25
44. IF (CHAR(I).EQ.' ') GO TO 10
45. ID = ' '
46. DO 20 J = 1,6
47. FLD (BIT,6,ID) = FLD (0,6,CHAR(I))
48. BIT = BIT + 6
49. I = I + 1
50. IF (CHAR(I).EQ.' ') RETURN
51. 20 CONTINUE
52. RETURN
53. END
```

DELTA, L N, INPUT, INPUT

```

1. C SUBROUTINE INPUT (FIRST)
2. C
3. C INPUT IS THE PROGRAM FOR READING IN THE RETRIEVAL REQUEST
4. C SELECTION CRITERIA FOR THE NAVDAB RETRIEVAL SYSTEM. THE
5. C DATA IS READ INTO COMMON /RETRV/. THIS INPUT PROGRAM IS
6. C FOR VERSION 1 MOD 1 OF NAVDAB.
7. C
8. C
9. C
10. C IMPLICIT INTEGER (A-Z)
11. C
12. C LOGICAL FIRST, BACKUP, CLSREQ
13. C
14. C REAL AEQUIV(4,10), EEQUIV(4,10), WORD(20)
15. C
16. C INCLUDE RETREV, LIST
17. C
18. C DIMENSION TEMP(15), UNITS(4), CHAR(81), MSQRS(2), NAME(10),
19. C IWORD(20), YEARS(2), DAYS(2), HOURS(2), ECSR(4), ENRCK(1)/999,99,
20. C 299,999/
21. C
22. C EQUIVALENCE (AEQUIV, ACPARS), (EEQUIV, ENPARS), (MSQRS(1), PMSQR),
23. C (MSQRS(2), TMSQR), (YEARS(1), FMYR), (YEARS(2), TOYR), (DAYS(1), FDAY),
24. C 2(DAYS(2), TODAY), (HOURS(1), FHTM), (HOURS(2), TOTM), (ECSR, EXP)
25. C
26. C DATA R /'R'/
27. C
28. C DATA W /'W'/
29. C
30. C BACKUP = .FALSE.
31. C CALL CLASS (NOSE, SEX, MASK, REQUEST, TNAME, TNBR, FIRST, BACKUP, CLSREQ)
32. C BACKUP = .TRUE.
33. C
34. C ENTER RETRIEVAL MODE
35. C
36. 9000 WRITE (6,9010)
37. 9010 FORMAT ('ENTER DESIRED DATA TYPE: /4X, 'ACOUSTIC ONLY('A'), ENVI
38. C RONMENTAL ONLY('E'), OR BOTH('B'),)
39. C READ (5,52,END=110,ERR=9030) MODE
40. C IF (MODE.EQ.'B'.OR.MODE.EQ.'E') GO TO 9040
41. C IF (MODE.EQ.'E') GO TO 9020
42. C IF (MODE.NE.'A') GO TO 9035
43. C RUNTYP = 1
44. C WRITE (6,9015)
45. 9015 FORMAT (5X, 'ACOUSTIC ONLY')
46. C HENDCR = 0
47. C NENPAR = 0
48. C NENTHD = 0
49. C GO TO 6
50. 9020 RUNTYP = 2
51. C WRITE (6,9025)
52. 9025 FORMAT (5X, 'ENVIRONMENTAL ONLY')
53. C HACDCR = 0
54. C NACPAR = 0

```

```

55.  IANTHD = 0
56.  NSTYPE = 7
57.  NSTYPE = 7
58.  HSSYST = 0
59.  HRSYST = 0
60.  GO TO 6
61.  9040 RUNTP = 3
62.  WRITE (A,9040)
63.  9045 FORMAT (5X,'ACOUSTIC AND ENVIRONMENTAL')
64.  GO TO 6
65.  C
66.  C
67.  C
68.  9030 WRITE (A,156)
69.  GO TO 9000
70.  9035 WRITE (A,9036)
71.  9036 FORMAT (' *** INVALID DATA TYPE SPECIFIED ***')
72.  GO TO 9000
73.  C
74.  C
75.  C
76.  C
77.  6
78.  7
79.  FORMAT ('* ENTER EXP. NO., CRUISE NO., STATION NO., RUN NO., RESTRICT')
80.  10N / ENTER ALL')
81.  READ (5,70,END=9030,ERR=8) (CHAR(I),I=1,80)
82.  WRITE (A,12) (CHAR(I),I=1,80)
83.  EXP = -99999999
84.  CRU = -99999999
85.  STA = -99999999
86.  RUN = -99999999
87.  CALL FREEED (CHAR,WORD,EXP,NWORDS,4,88)
88.  IF (NWORDS.GT.0.AND.CHAR(1).NE.'A') GO TO 3
89.  EXP = -99999999
90.  GO TO 9030
91.  C
92.  C
93.  C
94.  3
95.  9
96.  8
97.  6
98.  11
99.  6
100.  C
101.  C
102.  C
103.  4910 WRITE (A,4920)
104.  4920 FORMAT ('* HAVE ALL SPECIFICATIONS BEEN ENTERED?')
105.  READ (5,70,END=6,ERR=9921) (CHAR(I),I=1,80)
106.  WRITE (A,12) (CHAR(I),I=1,80)
107.  IF (CHAR(1).EQ.'Y') GO TO 4925
108.  IF (CHAR(1).EQ.' 'OR.CHAR(1).EQ.'N') GO TO 4930
109.  C
110.  C
111.  C

```



```

112. WRITE (6,4)
113. FORMAT (1,000 INVALID RESPONSE TO YES OR NO QUESTION ****)
114. GO TO 4912
115. 4921 WRITE (6,50)
116. GO TO 4910
117.
118. 4925 ANY = 0
119. :LAT = 0
120. ELON = 0
121. SLAT = 0
122. WLOM = 0
123. FMYR = 0
124. TOYR = 0
125. NMOM = 2
126. FMDAY = 0
127. TODAY = 0
128. FMTIM = 0
129. TOTIM = 0
130. NACDCR = 0
131. NACPAR = 0
132. NAMTHD = 0
133. MENDCR = 0
134. NENPAR = 0
135. NEMTHD = 0
136. NSTYPE = 0
137. NRKTYPE = 0
138. NSSYST = 0
139. NRSYST = 0
140. GO TO 900
141. C
142. C ENTER AREA
143. C
144. 4930 ANY = 1
145. 5 PRINT 10
146. 10 FORMAT (/,1 ENTER WHOLE WORLD('''), RECTANGULAR AREA('''), OR
147. INSQR AREA('''),
148. READ (5,70,END=0,ERR=43) (CHAR(I),I=1,80)
149. WRITE (6,12) (CHAR(I),I=1,80)
150. FORMAT (3X,80A1)
151. CALL FREEFD (CHAR,WORD,MSQRS,MHSQRS,2,843)
152. IF (MHSQRS.LT.1) GO TO 43
153. IF (R.EQ.MHSQRS(1)) GO TO 40 W RECTANGULAR AREA
154. IF (W.EQ.MHSQRS(1)) GO TO 30 W WHOLE WORLD
155. C
156. C MSQR REQUEST
157. C
158. C IF (MHSQRS.EQ.1) TMSQR = FMSQR
159. C
160. IF (FMSQR.GT. 0 .AND. FMSQR.LT. 289) GO TO 14
161. IF (FMSQR.GT. 299 .AND. FMSQR.LT. 624) GO TO 14
162. IF (FMSQR.GT. 900 .AND. FMSQR.LT. 937) GO TO 14
163. GO TO 17
164. C
165. 16 IF (TMSQR.GT. 0 .AND. TMSQR.LT. 289) GO TO 25
166. IF (TMSQR.GT. 299 .AND. TMSQR.LT. 624) GO TO 25
167. IF (TMSQR.GT. 900 .AND. TMSQR.LT. 937) GO TO 25
168. C

```

```

169. C          ERRUR
170. C
171. 17 PRINT 19,FMSQR,TMSQR
172. 19 FORMAT (1,'... ERROR-- MSQR ',J3,' OR ',J3,' IS ILLEGAL ....')
173. GO TO 5
174. C
175. C          CONVERT TO LONG & LAT
176. C
177. 25 CALL MSQCVT (FMSQR,TMSQR,NLAT,ELON,SLAT,MLON)
178. C
179. C          GO TO NEXT INPUT ITEM
180. C
181. C          GO TO 50
182. C
183. C          WHOLE WORLD
184. C
185. 30 NLAT = 0
186. ELON = 0
187. SLAT = 0
188. MLON = 0
189. GO TO 50
190. C
191. C          RECTANGULAR AREA
192. C
193. C          NORTHERN BOUNDARY
194. C
195. 40 WRITE (6,41)
196. 41 FORMAT (1,' UPPER BOUNDARY!')
197. C
198. NLATNS = , ,
199. READ (5,70,END=5,ERR=46) (CHAR(1),I=1,80)
200. CALL FREEFD (CHAR,WORD,INORD,NWORDS,3,546)
201. IF (NWORDS.EQ.0) GO TO 46
202. IF (INORD(1).EQ.-99999999) INORD(1) = 0
203. NLATD = INORD(1)
204. IF (NWORDS.EQ.3) GO TO 42
205. NLATM = 0
206. FLD(0,6,NLATNS) = FLD(0,6,INORD(2))
207. GO TO 5000
208. 42 IF (WORD(2).EQ.-99999999.) WORD(2) = 0.
209. NLATM = WORD(2) + 10.
210. FLD(0,6,NLATNS) = FLD(0,6,INORD(3))
211. C
212. C          CHECK FOR ERROR
213. C
214. 5000 IF (NLATD .LT. 0 .OR. NLATD .GT. 90) GO TO 46
215. IF (NLATM .LT. 0 .OR. NLATM .GT. 599) GO TO 46
216. IF (NLATNS .NE. 'N' .AND. NLATNS .NE. 'S') GO TO 46
217. NLAT = NLATD + 1000 + NLATM
218. IF (NLATNS .EQ. 'S') NLAT = -NLAT
219. C
220. C          SOUTHERN BOUNDARY
221. C
222. 59 WRITE (6,60)
223. 60 FORMAT (1,' LOWER BOUNDARY!')
224. C
225. SLATNS = , ,
226. READ (5,70,END=40,ERR=47) (CHAR(1),I=1,80)
227. CALL FREEFD (CHAR,WORD,INORD,NWORDS,3,547)

```

```

226. IF (NORDS.EQ.0) GO TO 47
227. IF (IWORD(1).EQ.-99999999) IWORD(1) = 0
228. SLAT0 = IWORD(1)
229. IF (NORDS.EQ.3) GO TO 61
230. SLATM = 0
231. FLD(2,6,SLATNS) = FLD(2,6,IWORD(2))
232. GO TO 5100
233. 61 IF (IWORD(2).EQ.-99999999) IWORD(2) = 0.
234. SLATM = IWORD(2) + 10.
235. FLD(2,6,SLATNS) = FLD(2,6,IWORD(3))
236. C
237. C CHECK FOR ERROR
238. C
239. C
240. 5100 IF (SLATD.LT.0 .OR. SLATD.GT.90) GO TO 47
241. IF (SLATM.LT.0 .OR. SLATM.GT.599) GO TO 47
242. IF (SLATNS.NE.'M'.AND. SLATNS.NE.'S') GO TO 47
243. SLAT = SLATD + 1000 + SLATM
244. IF (SLATNS.EQ.'S') SLAT = -SLAT
245. IF (SLAT.GT.NLAT) GO TO 46
246. C
247. C WESTERN BOUNDARY
248. C
249. 65 WRITE (6,66)
250. 66 FORMAT (1X LEFT BOUNDARY:1)
251. BLONEM = 1
252. READ (5,70,END=59,ERR=49) (CHAR(I),I=1,80)
253. CALL FREEFD (CHAR,WORD,IWORD,NORDS,3,598)
254. IF (NORDS.EQ.0) GO TO 49
255. IF (IWORD(1).EQ.-99999999) IWORD(1) = 0
256. BLOND = IWORD(1)
257. IF (NORDS.EQ.3) GO TO 67
258. BLONM = 0
259. FLD(2,6,WLONEM) = FLD(2,6,IWORD(2))
260. GO TO 5300
261. 67 IF (IWORD(2).EQ.-99999999) IWORD(2) = 0.
262. WLONM = IWORD(2) + 10.
263. FLD(2,6,WLONEM) = FLD(2,6,IWORD(3))
264. C
265. C CHECK FOR ERROR
266. C
267. 5300 IF (WLOND.LT.0 .OR. WLOND.GT.180) GO TO 49
268. IF (WLONM.LT.0 .OR. WLONM.GT.599) GO TO 49
269. IF (WLONEM.NE.'E'.AND. WLONEM.NE.'W') GO TO 49
270. WLON = WLOND + 1000 + WLONM
271. IF (WLONEM.EQ.'W') WLON = -WLON
272. C
273. C EASTERN BOUNDARY
274. C
275. 62 WRITE (6,63)
276. 63 FORMAT (1X RIGHT BOUNDARY:1)
277. BLONEM = 1
278. READ (5,70,END=45,ERR=48) (CHAR(I),I=1,80)
279. CALL FREEFD (CHAR,WORD,IWORD,NORDS,3,598)
280. IF (NORDS.EQ.0) GO TO 48
281. IF (IWORD(1).EQ.-99999999) IWORD(1) = 0
282. FLOND = IWORD(1)
283. IF (NORDS.EQ.3) GO TO 64

```

```

283. ELONM = 0
284. FLD(1,4,ELONM) = FLD(0,4,WORD(2))
285. GO TO 520C
286.
287. 44 IF (WORD(2).EQ.-999999999.) WORD(2) = 0.
288. ELONM = WORD(2) * 10.
289. FLD(1,4,ELONM) = FLD(0,4,WORD(3))
290.
291. C
292. C
293. C CHECK FOR ERROR
294. C
295. C
296. C
297. C
298. C
299. C
300. C
301. C
302. C
303. C
304. C
305. C
306. C
307. C
308. C
309. C
310. C
311. C
312. C
313. C
314. C
315. C
316. C
317. C
318. C
319. C
320. C
321. C
322. C
323. C
324. C
325. C
326. C
327. C
328. C
329. C
330. C
331. C
332. C
333. C
334. C
335. C
336. C
337. C
338. C
339. C
340. C
341. C
342. C
343. C
344. C
345. C
346. C
347. C
348. C
349. C
350. C
351. C
352. C
353. C
354. C
355. C
356. C
357. C
358. C
359. C
360. C
361. C
362. C
363. C
364. C
365. C
366. C
367. C
368. C
369. C
370. C
371. C
372. C
373. C
374. C
375. C
376. C
377. C
378. C
379. C
380. C
381. C
382. C
383. C
384. C
385. C
386. C
387. C
388. C
389. C
390. C
391. C
392. C
393. C
394. C
395. C
396. C
397. C
398. C
399. C
400. C
401. C
402. C
403. C
404. C
405. C
406. C
407. C
408. C
409. C
410. C
411. C
412. C
413. C
414. C
415. C
416. C
417. C
418. C
419. C
420. C
421. C
422. C
423. C
424. C
425. C
426. C
427. C
428. C
429. C
430. C
431. C
432. C
433. C
434. C
435. C
436. C
437. C
438. C
439. C
440. C
441. C
442. C
443. C
444. C
445. C
446. C
447. C
448. C
449. C
450. C
451. C
452. C
453. C
454. C
455. C
456. C
457. C
458. C
459. C
460. C
461. C
462. C
463. C
464. C
465. C
466. C
467. C
468. C
469. C
470. C
471. C
472. C
473. C
474. C
475. C
476. C
477. C
478. C
479. C
480. C
481. C
482. C
483. C
484. C
485. C
486. C
487. C
488. C
489. C
490. C
491. C
492. C
493. C
494. C
495. C
496. C
497. C
498. C
499. C
500. C
501. C
502. C
503. C
504. C
505. C
506. C
507. C
508. C
509. C
510. C
511. C
512. C
513. C
514. C
515. C
516. C
517. C
518. C
519. C
520. C
521. C
522. C
523. C
524. C
525. C
526. C
527. C
528. C
529. C
530. C
531. C
532. C
533. C
534. C
535. C
536. C
537. C
538. C
539. C
540. C
541. C
542. C
543. C
544. C
545. C
546. C
547. C
548. C
549. C
550. C
551. C
552. C
553. C
554. C
555. C
556. C
557. C
558. C
559. C
560. C
561. C
562. C
563. C
564. C
565. C
566. C
567. C
568. C
569. C
570. C
571. C
572. C
573. C
574. C
575. C
576. C
577. C
578. C
579. C
580. C
581. C
582. C
583. C
584. C
585. C
586. C
587. C
588. C
589. C
590. C
591. C
592. C
593. C
594. C
595. C
596. C
597. C
598. C
599. C
600. C
601. C
602. C
603. C
604. C
605. C
606. C
607. C
608. C
609. C
610. C
611. C
612. C
613. C
614. C
615. C
616. C
617. C
618. C
619. C
620. C
621. C
622. C
623. C
624. C
625. C
626. C
627. C
628. C
629. C
630. C
631. C
632. C
633. C
634. C
635. C
636. C
637. C
638. C
639. C
640. C
641. C
642. C
643. C
644. C
645. C
646. C
647. C
648. C
649. C
650. C
651. C
652. C
653. C
654. C
655. C
656. C
657. C
658. C
659. C
660. C
661. C
662. C
663. C
664. C
665. C
666. C
667. C
668. C
669. C
670. C
671. C
672. C
673. C
674. C
675. C
676. C
677. C
678. C
679. C
680. C
681. C
682. C
683. C
684. C
685. C
686. C
687. C
688. C
689. C
690. C
691. C
692. C
693. C
694. C
695. C
696. C
697. C
698. C
699. C
700. C
701. C
702. C
703. C
704. C
705. C
706. C
707. C
708. C
709. C
710. C
711. C
712. C
713. C
714. C
715. C
716. C
717. C
718. C
719. C
720. C
721. C
722. C
723. C
724. C
725. C
726. C
727. C
728. C
729. C
730. C
731. C
732. C
733. C
734. C
735. C
736. C
737. C
738. C
739. C
740. C
741. C
742. C
743. C
744. C
745. C
746. C
747. C
748. C
749. C
750. C
751. C
752. C
753. C
754. C
755. C
756. C
757. C
758. C
759. C
760. C
761. C
762. C
763. C
764. C
765. C
766. C
767. C
768. C
769. C
770. C
771. C
772. C
773. C
774. C
775. C
776. C
777. C
778. C
779. C
780. C
781. C
782. C
783. C
784. C
785. C
786. C
787. C
788. C
789. C
790. C
791. C
792. C
793. C
794. C
795. C
796. C
797. C
798. C
799. C
800. C
801. C
802. C
803. C
804. C
805. C
806. C
807. C
808. C
809. C
810. C
811. C
812. C
813. C
814. C
815. C
816. C
817. C
818. C
819. C
820. C
821. C
822. C
823. C
824. C
825. C
826. C
827. C
828. C
829. C
830. C
831. C
832. C
833. C
834. C
835. C
836. C
837. C
838. C
839. C
840. C
841. C
842. C
843. C
844. C
845. C
846. C
847. C
848. C
849. C
850. C
851. C
852. C
853. C
854. C
855. C
856. C
857. C
858. C
859. C
860. C
861. C
862. C
863. C
864. C
865. C
866. C
867. C
868. C
869. C
870. C
871. C
872. C
873. C
874. C
875. C
876. C
877. C
878. C
879. C
880. C
881. C
882. C
883. C
884. C
885. C
886. C
887. C
888. C
889. C
890. C
891. C
892. C
893. C
894. C
895. C
896. C
897. C
898. C
899. C
900. C
901. C
902. C
903. C
904. C
905. C
906. C
907. C
908. C
909. C
910. C
911. C
912. C
913. C
914. C
915. C
916. C
917. C
918. C
919. C
920. C
921. C
922. C
923. C
924. C
925. C
926. C
927. C
928. C
929. C
930. C
931. C
932. C
933. C
934. C
935. C
936. C
937. C
938. C
939. C
940. C
941. C
942. C
943. C
944. C
945. C
946. C
947. C
948. C
9
```



```

340. 52 FORMAT (A1)
341. IF (BOUND5.EQ.'Y') GO TO 53
342. IF (BOUND5.EQ.'N') GO TO 49
343. C
344. C
345. C
346. WRITE (6,4)
347. GO TO 50
348. 57 WRITE (6,56)
349. GO TO 50
350. C
351. C
352. C
353. 53 PRINT 54
354. 54 FORMAT (1, ENTER YEAR BOUNDARIES)
355. READ (5,70,END=53,ERR=55) (CHAN(1),1=1,80)
356. FMYR = 0
357. TOYR = 0
358. CALL FREEFD (CHAR,WORD,YEARS,NWORDS,2,555)
359. IF (NWORDS.EQ.0) GO TO 72
360. IF (NWORDS.EQ.1) TOYR = FMYR
361. 15 FORMAT (1)
362. C
363. C
364. C
365. CHECK FOR ERROR
366. IF (FMYR.LT.0 .OR. FMYR.GT.99) GO TO 39
367. IF (TOYR.LT.0 .OR. TOYR.GT.99) GO TO 58
368. PRINT 15,FMYR,TOYR
369. GO TO 72
370. C
371. C
372. C
373. C
374. 55 PRINT 56
375. 56 FORMAT (1, ... READ FORMAT ERROR ...)
376. GO TO 53
377. 39 WRITE (6,78) FMYR
378. GO TO 53
379. 58 WRITE (6,78) TOYR
380. GO TO 53
381. WRITE (6,89)
382. GO TO 53
383. C
384. C
385. C
386. C
387. C
388. C
389. 72 PRINT 71
390. 71 FORMAT (1, ENTER MONTH NUMBERS)
391. READ (5,70,ERR=75,END=53) (CHAN(1),1=1,80)
392. 70 FORMAT (80A1)
393. CALL FREEFD (CHAR,WORD,MONTHS,NMON,12,875)
394. IF (NMON.EQ.0) GO TO 80
395. C
396. C
397. C
398. C
399. C
400. C
401. C
402. C
403. C
404. C
405. C
406. C
407. C
408. C
409. C
410. C
411. C
412. C
413. C
414. C
415. C
416. C
417. C
418. C
419. C
420. C
421. C
422. C
423. C
424. C
425. C
426. C
427. C
428. C
429. C
430. C
431. C
432. C
433. C
434. C
435. C
436. C
437. C
438. C
439. C
440. C
441. C
442. C
443. C
444. C
445. C
446. C
447. C
448. C
449. C
450. C
451. C
452. C
453. C
454. C
455. C
456. C
457. C
458. C
459. C
460. C
461. C
462. C
463. C
464. C
465. C
466. C
467. C
468. C
469. C
470. C
471. C
472. C
473. C
474. C
475. C
476. C
477. C
478. C
479. C
480. C
481. C
482. C
483. C
484. C
485. C
486. C
487. C
488. C
489. C
490. C
491. C
492. C
493. C
494. C
495. C
496. C
497. C
498. C
499. C
500. C
501. C
502. C
503. C
504. C
505. C
506. C
507. C
508. C
509. C
510. C
511. C
512. C
513. C
514. C
515. C
516. C
517. C
518. C
519. C
520. C
521. C
522. C
523. C
524. C
525. C
526. C
527. C
528. C
529. C
530. C
531. C
532. C
533. C
534. C
535. C
536. C
537. C
538. C
539. C
540. C
541. C
542. C
543. C
544. C
545. C
546. C
547. C
548. C
549. C
550. C
551. C
552. C
553. C
554. C
555. C
556. C
557. C
558. C
559. C
560. C
561. C
562. C
563. C
564. C
565. C
566. C
567. C
568. C
569. C
570. C
571. C
572. C
573. C
574. C
575. C
576. C
577. C
578. C
579. C
580. C
581. C
582. C
583. C
584. C
585. C
586. C
587. C
588. C
589. C
590. C
591. C
592. C
593. C
594. C
595. C
596. C
597. C
598. C
599. C
600. C
601. C
602. C
603. C
604. C
605. C
606. C
607. C
608. C
609. C
610. C
611. C
612. C
613. C
614. C
615. C
616. C
617. C
618. C
619. C
620. C
621. C
622. C
623. C
624. C
625. C
626. C
627. C
628. C
629. C
630. C
631. C
632. C
633. C
634. C
635. C
636. C
637. C
638. C
639. C
640. C
641. C
642. C
643. C
644. C
645. C
646. C
647. C
648. C
649. C
650. C
651. C
652. C
653. C
654. C
655. C
656. C
657. C
658. C
659. C
660. C
661. C
662. C
663. C
664. C
665. C
666. C
667. C
668. C
669. C
670. C
671. C
672. C
673. C
674. C
675. C
676. C
677. C
678. C
679. C
680. C
681. C
682. C
683. C
684. C
685. C
686. C
687. C
688. C
689. C
690. C
691. C
692. C
693. C
694. C
695. C
696. C
697. C
698. C
699. C
700. C
701. C
702. C
703. C
704. C
705. C
706. C
707. C
708. C
709. C
710. C
711. C
712. C
713. C
714. C
715. C
716. C
717. C
718. C
719. C
720. C
721. C
722. C
723. C
724. C
725. C
726. C
727. C
728. C
729. C
730. C
731. C
732. C
733. C
734. C
735. C
736. C
737. C
738. C
739. C
740. C
741. C
742. C
743. C
744. C
745. C
746. C
747. C
748. C
749. C
750. C
751. C
752. C
753. C
754. C
755. C
756. C
757. C
758. C
759. C
760. C
761. C
762. C
763. C
764. C
765. C
766. C
767. C
768. C
769. C
770. C
771. C
772. C
773. C
774. C
775. C
776. C
777. C
778. C
779. C
780. C
781. C
782. C
783. C
784. C
785. C
786. C
787. C
788. C
789. C
790. C
791. C
792. C
793. C
794. C
795. C
796. C
797. C
798. C
799. C
800. C
801. C
802. C
803. C
804. C
805. C
806. C
807. C
808. C
809. C
810. C
811. C
812. C
813. C
814. C
815. C
816. C
817. C
818. C
819. C
820. C
821. C
822. C
823. C
824. C
825. C
826. C
827. C
828. C
829. C
830. C
831. C
832. C
833. C
834. C
835. C
836. C
837. C
838. C
839. C
840. C
841. C
842. C
843. C
844. C
845. C
846. C
847. C
848. C
849. C
850. C
851. C
852. C
853. C
854. C
855. C
856. C
857. C
858. C
859. C
860. C
861. C
862. C
863. C
864. C
865. C
866. C
867. C
868. C
869. C
870. C
871. C
872. C
873. C
874. C
875. C
876. C
877. C
878. C
879. C
880. C
881. C
882. C
883. C
884. C
885. C
886. C
887. C
888. C
889. C
890. C
891. C
892. C
893. C
894. C
895. C
896. C
897. C
898. C
899. C
900. C
901. C
902. C
903. C
904. C
905. C
906. C
907. C
908. C
909. C
910. C
911. C
912. C
913. C
914. C
915. C
916. C
917. C
918. C
919. C
920. C
921. C
922. C
923. C
924. C
925. C
926. C
927. C
928. C
929. C
930. C
931. C
932. C
933. C
934. C
935. C
936. C
937. C
938. C
939. C
940. C
941. C
942. C
943. C
944. C
945. C
946. C
947. C
948. C
949. C
950. C
951. C
952. C
953. C
954. C
955. C
956. C
957. C
958. C
959. C
960. C
961. C
962. C
963. C
964. C
965. C
966. C
967. C
968. C
969. C
970. C
971. C
972. C
973. C
974. C
975. C
976. C
977. C
978. C
979. C
980. C
981. C
982. C
983. C
984. C
985. C
986. C
987. C
988. C
989. C
990. C
991. C
992. C
993. C
994. C
995. C
996. C
997. C
998. C
999. C

```

```

397. IF (MONTHS(1) .LT. 1 .OR. MONTHS(1) .GT. 12) GO TO 77
398. 73 CONTINUE
399. C
400. C PRINT
401. C
402. PRINT 74, (MONTHS(1), 1-1, MMON)
403. 74 FORMAT (1X, 14, 10(' ', 14))
404. GO TO 80
405. C
406. C ERROR
407. C
408. 75 PRINT 54
409. GO TO 72
410. 77 WRITE (6, 78) MONTHS(1)
411. 78 FORMAT ('... ENTERED NUMBER', 15, ' OUT OF RANGE ....')
412. GO TO 72
413. C
414. C
415. C READ IN DAY BOUNDARIES
416. C
417. 80 PRINT 81
418. 81 FORMAT (' ENTER DAY BOUNDARIES:')
419. READ (5, 70, END=72, ERR=85) (CHAR(1), 1-1, 80)
420. FMDAY = 0
421. TODAY = 0
422. CALL FREED (CHAR, WORD, DAYS, MMWORDS, 2, 185)
423. IF (MMWORDS.EQ.0) GO TO 90
424. IF (MMWORDS.EQ.1) TODAY = FMDAY
425. C
426. C CHECK FOR ERROR
427. C
428. C
429. IF (FMDAY .EQ. 0 .AND. TODAY .EQ. 0) GO TO 83
430. IF (FMDAY .LT. 1 .OR. FMDAY .GT. 31) GO TO 86
431. IF (TODAY .LT. 1 .OR. TODAY .GT. 31) GO TO 87
432. IF (TODAY .LT. FMDAY) GO TO 88
433. C
434. C PRINT
435. 83 PRINT 15, FMDAY, TODAY
436. GO TO 90
437. C
438. C ERROR
439. C
440. 85 PRINT 54
441. GO TO 80
442. 86 WRITE (6, 78) FMDAY
443. GO TO 80
444. 87 WRITE (6, 78) TODAY
445. GO TO 80
446. 88 WRITE (6, 89)
447. 89 FORMAT ('... INVALID RANGE SPECIFIED ....')
448. GO TO 80
449. C
450. C
451. C READ IN TIME BOUNDARIES
452. C
453. 90 PRINT 91

```

```

454. 91 FORMAT (1, ENTER TIME BOUNDARIES,1)
455. READ (5,70,END=80,ERR=95) (CHAR(I),I=1,80)
456. FMTIM = 0
457. TOTIM = 0
458. CALL FREEFD (CHAR,4,ORD,HOURS,NOORDS,2,595)
459. IF (NOORDS.EQ.0) GO TO 120
460. IF (NOORDS.EQ.1) TOTIM = FMTIM
461. C
462. C CHECK FOR ERROR
463. C
464. C
465. IF (FMTIM.LT.0.OR.FMTIM.GT.2359) GO TO 96
466. IF (TOTIM.LT.0.OR.TOTIM.GT.2359) GO TO 97
467. C
468. C PRINT
469. C
470. PRINT 15,FMTIM,TOTIM
471. GO TO 120
472. C
473. C ERROR
474. C
475. C 95 PRINT 56
476. GO TO 90
477. C
478. 96 WRITE (6,78) FMTIM
479. GO TO 90
480. C
481. 97 WRITE (6,78) TOTIM
482. GO TO 90
483. C
484. C BOUNDARIES EQUAL ZERO
485. C
486. FMYR = 0
487. TOYR = 0
488. MMON = 0
489. FMOAY = 0
490. TODAY = 0
491. FMTIM = 0
492. TOTIM = 0
493. PRINT 103
494. 103 FORMAT (1, YEAR,MONTH,DAY,AND TIME BOUNDARIES WILL BE IGNORED,1)
495. C
496. C
497. C PROCESS ACOUSTIC REQUEST UNLESS ENV. ONLY
498. C
499. C
500. 120 IF (RNTYP.EQ.2) GO TO 212
501. PRINT 121
502. 121 FORMAT (10 ACOUSTIC REQUEST(S))
503. C
504. C BYPASS SOURCE/RECEIVER QUESTIONS
505. C
506. C
507. 1110 WRITE (6,1120)
508. 1120 FORMAT (1,QDO YOU WANT TO ENTER TRANSDUCER TYPE(S) OR SONAR SYSTEM(
509. 1517))
510. C
511. HEAD (5,1130,END=1158,ERR=1150) ANSWER
512. C
513. 1130 FORMAT (A3)
514. C
515. 1140 WRITE (6,1140)
516. 1140 FORMAT (3A,A3)
517. C
518. C
519. FLD (0.6,K) = FLD (0.6,ANSWER)
520. C

```

```

511. IF (A.EQ.'Y') GO TO 1151
512. NSTYPE = C
513. NSTYPE = C
514. NSTYPE = C
515. NSTYPE = C
516. NSTYPE = 0
517. IF (A.EQ.'..00..00..00') GO TO 135
518. C
519. C
520. C
521. WRITE (4,1)
522. GO TO 1110
523. C
524. C
525. C
526. 1151 PRINT 1152
527. 1152 FORMAT (1'DENTER SOURCE TYPE(S)')
528. READ (5,70.END=1110,ERR=1155) (CHAR(1),1=1,80)
529. CALL FREEED (CHAR,WORD,NSTYPE,10,1155)
530. IF (NSTYPE.EQ.0) GO TO 1161
531. C
532. C
533. C
534. C
535. C
536. C
537. C
538. C
539. C
540. DO 1153 I = 1,NSTYPE
541. IF (ISTYPE(I).LT.1.OR.SIYPE(I).GT.99) GO TO 1157
542. 1153 CONTINUE
543. C
544. C
545. C
546. C
547. C
548. C
549. C
550. C
551. C
552. C
553. C
554. C
555. C
556. C
557. C
558. C
559. C
560. C
561. C
562. C
563. C
564. C
565. C
566. C
567. C

```



```

568. 1163 CONTINUE
569. C
570. C PRINT
571. C
572. DO 1164 I = 1, NRTYPE
573. CALL NAMEIT (5, RTYPE(I), NAME, NR)
574. 1164 WRITE (6, 133) RTYPE(I), NAME(J), J = 1, NR
575. GO TO 1171
576. C
577. C ERROR
578. C
579. 1165 PRINT 56
580. GO TO 1161
581. 1167 WRITE (6, 78) RTYPE(I)
582. GO TO 1161
583. C
584. C READ IN SOURCE SYSTEMS
585. C
586. 1171 PRINT 1172
587. 1172 FORMAT('CENTER SOURCE SYSTEM(S)')
588. READ (5, 70, END = 1161, ERR = 1175) (CHAR(I), I = 1, 80)
589. CALL FREEED (CHAR, WORD, SSYST, NSYST, 10, 81175)
590. IF (NSYST.EQ.0) GO TO 1161
591. C
592. C CHECK FOR ERROR
593. C
594. 1174 DO 1173 I = 1, NSYST
595. IF (SSYST(I).LT.1.OR.SSYST(I).GT.99) GO TO 1177
596. 1173 CONTINUE
597. C
598. C PRINT
599. C
600. DO 1176 I = 1, NSYST
601. CALL NAMEIT (4, SSYST(I), NAME, NR)
602. 1176 WRITE (6, 139) SSYST(I), NAME(J), J = 1, NR
603. GO TO 1181
604. C
605. C ERROR
606. C
607. 1175 PRINT 56
608. GO TO 1171
609. 1177 WRITE (6, 78) SSYST(I)
610. GO TO 1171
611. C
612. C READ IN RECEIVER SYSTEMS
613. C
614. 1181 PRINT 1182
615. 1182 FORMAT('CENTER RECEIVER SYSTEM(S)')
616. READ (5, 70, END = 1171, ERR = 1185) (CHAR(I), I = 1, 80)
617. CALL FREEED (CHAR, WORD, MSYST, NMSYST, 10, 81185)
618. IF (NMSYST.EQ.0) GO TO 135
619. C
620. C CHECK FOR ERROR
621. C
622. 1184 DO 1183 I = 1, NMSYST
623. IF (MSYST(I).LT.1.OR.MSYST(I).GT.99) GO TO 1187
624. 1183 CONTINUE

```

```

625. C
626. C PRINT
627. C
628. DO 1186 I = 1,NRYSYST
629. CALL NAMEIT (4,RSYST(I),NAME,NR)
630. WRITE (6,139) RSYST(I),NAME(I),J=1,NR
631. GO TO 135
632. C
633. C ERROR
634. C
635. 1185 PRINT 56
636. GO TO 1181
637. 1187 WRITE (6,78) RSYST(I)
638. GO TO 1181
639. C
640. C READ IN ACOUSTIC DESCRIPTORS
641. C
642. C
643. 135 PRINT 136
644. 138 FORMAT ('CENTER ACOUSTIC DESCRIPTOR(S)')
645. READ (5,70,ERR=140,END=145) (CHAR(I),I=1,80)
646. CALL FREEED (CHAR,WORD,ACDCRS,NACDCR,10,5140)
647. IF (NACDCR.EQ.0) GO TO 140
648. C
649. C CHECK FOR ERROR
650. C
651. 138 DO 137 I=1,NACDCR
652. IF (ACDCRS(I)) .LT. 1 .OR. ACDCRS(I) .GT. 99) GO TO 142 @ TABLE LIMIT 99
653. 137 CONTINUE
654. C
655. C PRINT
656. C
657. DO 141 I = 1,NACDCR
658. CALL NAMEIT (8,ACDCRS(I),NAME,NR)
659. WRITE (6,139) ACDCRS(I),NAME(I),J=1,NR
660. 139 FORMAT (1X,11, ' ',10A4)
661. CONTINUE
662. GO TO 140
663. C
664. C ERROR
665. C
666. 140 PRINT 56
667. GO TO 135
668. 142 WRITE (6,78) ACDCRS(I)
669. GO TO 135
670. C
671. C BACKUP
672. C
673. 145 IF (N.EQ.'Y') GO TO 1181
674. GO TO 1110
675. C
676. C
677. C HEAD ACOUSTIC PARAMETERS
678. C
679. C
680. 140 PRINT 161
681. 161 FORMAT ('CENTER ACOUSTIC PARAMETERS (NAME,MIN,MAX,UNIT,METHODS) /

```

```

482.      C      ENTER 'END'
483.      C      NACPAR = C
484.      C      LOOP TO HEAD PARAMETERS
485.      C
486.      C      165 NACPAR = NACPAR + 1
487.      167 PRINT 170,NACPAR
488.      170 FORMAT (10PARAMETER ',12)
489.      READ (5,70,END=177,ERR=175) (CHAR(I),I=1,80)
490.      IF (CHAR(1)).EQ.'E' GO TO 180
491.      C
492.      C      CHECK FOR PARAMETER ARRAY LIMIT
493.      C
494.      C
495.      IF (NACPAR.LE.10) GO TO 174
496.      WRITE (6,173)
497.      173 FORMAT (1' *** INPUT RESTRICTED TO 10 ACOUSTIC PARAMETERS ***')
498.      GO TO 181
499.      C
500.      C      STORE MIN, MAX, AND UNIT VALUE IN CASE NOT SPECIFIED
501.      C
502.      174 WORD(2) = -99999999.
503.      WORD(3) = -99999999.
504.      WORD(4) = 0
505.      CALL FREED (CHAR,WORD,1,WORD,NWORDS,14,5175)
506.      IF (NWORDS.EQ.0) GO TO 175
507.      ACARS(1,NACPAR) = 1,WORD(1)
508.      Aequiv(2,NACPAR) = WORD(2)
509.      Aequiv(3,NACPAR) = WORD(3)
510.      IF (1,WORD(4).EQ.-99999999) 1,WORD(4) = 0
511.      ACARS(4,NACPAR) = 1,WORD(4)
512.      NAMTHD(NACPAR) = NWORDS - 4
513.      IF (NAMTHD(NACPAR).LT.0) NAMTHD(NACPAR) = 0
514.      C
515.      C      CHECK FOR ERRORS
516.      C
517.      C
518.      IF (ACARS(1,NACPAR).LT.1.OR.ACARS(1,NACPAR).GT.72) GO TO 179
519.      DUM = 0
520.      CALL CONVAT (DUM,ACARS(1,NACPAR),ACARS(4,NACPAR),UNITS)
521.      IF (UNITS(1).EQ.'UNITS') GO TO 182
522.      C
523.      C      PRINT PARAMETER INFORMATION
524.      C
525.      CALL NAMEST (ACARS(1,NACPAR),TEMP)
526.      IF (Aequiv(2,NACPAR).EQ.-99999999) GO TO 198
527.      IF (Aequiv(3,NACPAR).EQ.-99999999) GO TO 196
528.      PRINT 195,TEMP(1),TEMP(2),ACARS(2,NACPAR),ACARS(3,NACPAR),
529.      *      UNITS(1),UNITS(2)
530.      195 FORMAT (1X,2A6,' BETWEEN ',E15.5,' AND ',E15.5,' IN ',2A6)
531.      GO TO 202
532.      196 PRINT 197, TEMP(1),TEMP(2),ACARS(2,NACPAR),UNITS(1),UNITS(2)
533.      197 FORMAT (1X,2A6,' GREATER THAN ',E15.5,' IN ',2A6)
534.      GO TO 202
535.      198 IF (Aequiv(3,NACPAR).EQ.-99999999) GO TO 200
536.      PRINT 199, TEMP(1),TEMP(2),ACARS(3,NACPAR),UNITS(1),UNITS(2)
537.      199 FORMAT (1X,2A6,' LESS THAN ',E15.5,' IN ',2A6)
538.      GO TO 202

```

```

739. 200 PRINT 201, TEMP(1),TEMP(2),UNITS(1),UNITS(2)
740. 201 FORMAT (12,2A6,' IN ',2A6)
741. C
742. C
743. C
744. 202 N = NMETHOD(NACPAR)
745. IF (N.EQ.0) GO TO 165
746. DO 171 J = 1,N
747. AMTHDS(1,NACPAR) = 14ORD(1+4)
748. 171 IF (AMTHDS(1,NACPAR).LT.1.0R.AMTHDS(1,NACPAR).GT.999) GO TO 176
749. PRINT 172, (AMTHDS(1,NACPAR),I=1,N)
750. 172 FORMAT (1, METHODS: ',14,10(1,'.14)')
751. GO TO 165
752. C
753. C
754. C
755. 175 PRINT 56
756. GO TO 167
757. 176 PRINT 178
758. 178 FORMAT (1, ... ERROR IN METHODS SPECIFICATION ...')
759. GO TO 167
760. 179 WRITE (6,28) ACPARS(1,NACPAR)
761. GO TO 167
762. 182 WRITE (6,183) 14ORD(14)
763. 183 FORMAT (1, ... UNITS CODE',15,' IS INVALID ...')
764. GO TO 167
765. C
766. 177 IF (NACPAR.EQ.1) GO TO 135
767. NACPAR = NACPAR - 1
768. GO TO 167
769. C
770. 180 PRINT 12, (CHAR(1),I=1,3)
771. 181 NACPAR = NACPAR - 1
772. C
773. C
774. C
775. C
776. C
777. 210 IF (RNTYP.EQ.1) GO TO 900
778. 212 PRINT 211
779. 211 FORMAT (10 ENVIRONMENTAL REQUEST(S)')
780. C
781. C
782. C
783. C
784. 235 PRINT 236
785. 236 FORMAT (10CENTER ENVIRONMENTAL DESCRIPTOR(S)')
786. READ (5,70,ERR=240,END=243) (CHAR(1),I=1,80)
787. CALL FREEED (CHAR,WORD,ENDOCR,ENDOCR,10,8240)
788. IF (NENDCR.EQ.0) GO TO 260
789. C
790. C
791. C
792. 238 DO 237 I=1,NENDCR
793. IF (ENDCRS(1).LT.1.0R.ENDCRS(1).GT.99) GO TO 242 W TABLE LIMITY 99
794. 237 CONTINUE
795. C

```



```

796. C PRINT
797. C
798. DO 239 I = 1, NENDCR
799. CALL NAMEIT (7, ENDCRS(I), NAME, NR)
800. WRITE (6, 139) ENDCRS(I), (NAME(J), J=1, NR)
801. 239 CONTINUE
802. GO TO 26C
803. C
804. C ERROR
805. C
806. 240 PRINT 54
807. GO TO 235
808. 242 WRITE (6, 78) ENDCRS(I)
809. GO TO 235
810. C
811. C BACKUP
812. C
813. 243 IF (RUNITP.EQ.3) GO TO 160
814. IF (BOUNDS.EQ.'Y') GO TO 90
815. GO TO 50
816. C
817. C
818. C
819. C
820. C
821. 260 PRINT 261
822. 261 FORMAT ('ENTER ENVIRONMENTAL PARAMETERS (NAME, MIN, MAX, UNIT, METHOD
823. 15) / ENTER 'END')
824. NENPAR = 0
825. C
826. C
827. C LOOP TO READ THEM
828. 265 NENPAR = NENPAR + 1
829. 267 PRINT 170, NENPAR
830. READ (5, 70, END=277, ERR=275) (CHAR(I), I=1, 80)
831. IF (CHAR(1).EQ.'E') GO TO 280
832. C
833. C CHECK FOR PARAMETER ARRAY LIMIT
834. C
835. IF (NENPAR.LE.10) GO TO 274
836. WRITE (6, 273)
837. 273 FORMAT ('... INPUT RESTRICTED TO 10 ENVIRONMENTAL PARAMETERS ...')
838. 1)
839. GO TO 281
840. C
841. 274 WORD(2) = -999999999.
842. WORD(3) = -999999999.
843. WORD(4) = 0
844. CALL FREEFD (CHAR, WORD, (WORD, NWORDS, 14, 8275)
845. IF (NWORDS.EQ.0) GO TO 275
846. ENPAR(1, NENPAR) = WORD(1)
847. EQUIV(2, NENPAR) = WORD(2)
848. EQUIV(3, NENPAR) = WORD(3)
849. IF (WORD(4).EQ.-99999999) (WORD(4) = 0
850. ENPAR(4, NENPAR) = WORD(4)
851. NENTHD(NENPAR) = NWORDS - 4
852. IF (NENTHD(NENPAR).LT.0) NENTHD(NENPAR) = 0

```

```

853. C
854. C
855. C
856. C
857. IF (ENPARS(1,NENPAR).LT.101.OR.ENPARS(1,NENPAR).GT.172) GO TO 279
858. DUM = 0
859. CALL CONVRT (DUM,ENPARS(1,NENPAR),-ENPARS(4,NENPAR),UNITS)
860. IF (UNITS(1).EQ. 'UNITS ') GO TO 282
861. C
862. C
863. C
864. C
865. CALL NAMEST (ENPARS(1,NENPAR),TEMP)
866. IF (LEQUIV(2,NENPAR).EQ.-99999999.) GO TO 269
867. IF (LEQUIV(3,NENPAR).EQ.-99999999.) GO TO 268
868. PRINT 195,TEMP(1),TEMP(2),ENPARS(2,NENPAR),ENPARS(3,NENPAR),
      * UNITS(1),UNITS(2)
869. GO TO 272
870. 268 PRINT 197, TEMP(1),TEMP(2),ENPARS(2,NENPAR),UNITS(1),UNITS(2)
871. GO TO 272
872. 269 IF (LEQUIV(3,NENPAR).EQ.-99999999.) GO TO 270
873. PRINT 199, TEMP(1),TEMP(2),ENPARS(3,NENPAR),UNITS(1),UNITS(2)
874. GO TO 272
875. 270 PRINT 201, TEMP(1),TEMP(2),UNITS(1),UNITS(2)
876. C
877. C
878. C
879. 272 N = NEMTHD(NENPAR)
880. IF (N.EQ.0) GO TO 265
881. DO 271 I = 1,N
882. EMTHDS(1,NENPAR) = IWORD(1,4)
883. 271 IF (EMTHOS(1,NENPAR).LT.1.OR.EMTHOS(1,NENPAR).GT.999) GO TO 276
884. PRINT 172, (EMTHDS(1,NENPAR),I=1,N)
885. GO TO 265
886. C
887. C
888. C
889. 275 PRINT 56
890. GO TO 267
891. 276 PRINT 178
892. GO TO 267
893. C
894. 277 IF (NENPAR.EQ.1) GO TO 235
895. NENPAR = NENPAR - 1
896. GO TO 267
897. 279 WRITE (6,78) ENPARS(1,NENPAR)
898. GO TO 267
899. 282 WRITE (6,183) IWORD(4)
900. GO TO 267
901. C
902. C
903. 280 PRINT 12, (CHAR(1),I=1,3)
904. 281 NENPAR = NENPAR - 1
905. C
906. C
907. C
908. C
909. C

```

CHECK FOR METHODS ERRORS

ERROR

END OF ENVIRONMENTAL

910. 900 WRITE (4,910)
911. 910 FORMAT ('0... SEARCH IN PROGRESS ...')
912. RETURN
913. END

FOR S* N,INTENS,,INTENS

```
1. SUBROUTINE INTENSIVALUE,UNICODE,UNITS)
2. C
3. C TABLE OF INTENSITY LEVEL CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(3)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1,1,3)/74,75,76/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,1),1,2,3),1,3)/'WATTS','M002',
18. C 'WATTS/CM','M002','WTS/CM','M002/H20/
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(3)='WATTS/'
23. C UNITS(4)='M002'
24. C
25. C IF UNIC00=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF(UNIC00.EQ.0) UNIC00=74
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA (FACTOR(1),1,1,3)/100,1.004,1.004/
32. C
33. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
34. C
35. C UNITS(1)='UNITS'
36. C UNITS(2)='ERROR'
37. C
38. C IF CODE IS IN TABLE, PERFORM CONVERSION, ALSO, ENTER UNITS ALPHA
39. C CODE INTO 'UNITS'.
40. C
41. C IC00=IABS(UNIC00)
42. C DO 2 I=1,3
43. C IF(IC00.NE.ENTRY(1,I)) GO TO 2
44. C IF(UNIC00.GE.0) VALUE=VALUE*FACTOR(I)
45. C IF(UNIC00.LT.0) VALUE=VALUE/FACTOR(I)
46. C UNITS(1)=ENTRY(2,I)
47. C UNITS(2)=ENTRY(3,I)
48. C
49. C AT THIS POINT, CONVERSION IS COMPLETE.
50. C
51. C RETURN
52. C
53. C ELSE, CHECK REST OF TABLE.
54. C
```



```

55.      2 CONTINUE
56.      C
57.      C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
58.      C
59.      PRINT IOI,ICODE
60.      IOL FORMAT(' ERROR--UNIT CODE',I3,' NOT IN INTENSITY TABLE.')
```

```

61.      RETURN
62.      END
```

QFOR,S= N,INVTY,,INVTY

```

1. SUBROUTINE INVTY
2.
3. C ROUTINE TO PREPARE AN INVENTORY OF THE RUNS MEETING THE SELECTION
4. C CRITERIA. THE EXPERIMENT, CAUSE, STATION, AND RUN NUMBER FOR EACH
5. C RUN ARE PRINTED.
6.
7. IMPLICIT INTEGER (A-Z)
8. DIMENSION NAME(10)
9.
10. COMMON /RUN /PORA,RILAT,RINS,NFLAT,RFNS,RILON,RLEN,NFLON,RFEN,
11. RIMO,RIDAY,RIVR,RITIM,RIZON,RFMO,RFDAY,RFYR,RFIIM,
12. RFION,NAVCO,DTASRC,SYSSRC,SYSRCH,SRCTYP,RCATYP,
13. CLAS,DESCR,MAVDIR,MAVHT,MTUNIT,BAPPRO,PDMOUNT,
14. SEASTA,BNODIR,BNOVEL,VELUNT,SMLDIR,SMLMT,
15. SMTUNT,SMLPRO,SPDMNT,MEATHR,BOTOPH,DPHUNT,BTMSLP,
16. BATMY,INFO(9),RSTAT(2),RUNNO
17.
18. WRITE (6,5)
19. S FORMAT (11) INVENTORY OF RUNS SATISFYING SELECTION CRITERIA: //TIS,
20. 1'EXPER.,T40,'EXPER.',T50,'CRUISE',T58,'STATION',
21. 2T68,'RUN/T3',TYPE,T15,'NUMBER',T23,'EXPER. NAME',T39,
22. 3'REFERENCE',T50,'NUMBER',T58,'NUMBER',T67,'NUMBER//)
23. CALL GETINT
24. 10 CALL GETRUN (DATYPE,EXPNO,CRUNO,STANO,830)
25. CALL NAMEIT (1,EXPNO,NAME,NR)
26. NR1 = NR - 1
27. NR2 = NR - 3
28. IF (DATYPE.EQ.9) GO TO 20
29. WRITE (6,15) (NAME(I),I=NR1,NR),CRUNO,STANO,RUNNO,EXPNO,
30. 1(NAME(1),I=1,NR2)
31. 15 FORMAT (T39,1('A6,A2,T50,I4,T58,I4,T67,I4,T2,'ACOUSTIC',T15,I4,
32. 12X,JAG)
33. GO TO 10
34. 20 WRITE (6,21) (NAME(I),I=NR1,NR),CRUNO,STANO,RUNNO,EXPNO,
35. 1(NAME(1),I=1,NR2)
36. 21 FORMAT (T39,1('A6,A2,T50,I4,T58,I4,T67,I4,T2,'ENVIRONMENTAL',T15,
37. 14,2X,JAG)
38. GO TO 10
39. RETURN
40. END

```

QFOR,58 N,LLHEAR,,LINEAR

```
1. SUBROUTINE LINEAR(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF LINEAR MEASURE CONVERSION FACTORS,
4. C
5. C INTEGER ENTRY(3,12),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(12)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,12)/2,3,4,5,6,7,8,9,10,11,12,06/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,J),J=1,2,3),J=1,12)/'MILLIM','ETERS',
18. C 'CENTIM','ETERS', 'METERS',
19. C 'KILOM','ETERS', 'INCHES',
20. C 'FEET', 'YARDS',
21. C 'KILOTA','NDS', 'NAUT','MILES',
22. C 'STAT','MILES', 'FATHOM','S',
23. C 'MICRON','S',
24. C
25. C ENTER INTERNAL UNITS.
26. C
27. C UNITS(3)= 'METERS'
28. C UNITS(4)= ' '
29. C
30. C IF UNICODE = U, THEN SET CODE TO STANDARD UNITS.
31. C
32. C IF (UNICODE.EQ.0) UNICDE=-4
33. C
34. C SET CONVERSION FACTORS.
35. C
36. C DATA (FACTOR(1),1=1,12)/.00100,.0100,.100,1.00,3.28,5.40,0.050,2,
37. C .3048,0.600,.914,0.192,1.700,9.14,0.192,1.702,
38. C 1.852,1.409,150,1.828,0.360,1.00,0.9
39. C
40. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
41. C CODE INTO 'UNITS'.
42. C
43. C ICODE=IABS(UNICDE)
44. C DO 2 I=1,12
45. C IF (ICODE.NE.ENTRY(1,I)) GO TO 2
46. C IF (UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
47. C IF (UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
48. C DO 1 J=1,2
49. C UNITS(J)=ENTRY(J+1,I)
50. C 1 CONTINUE
51. C
52. C AT THIS POINT, CONVERSION IS COMPLETE.
53. C
54. C RETURN
```

```

55. C
56. C ELSE, CHECK REST OF TABLE.
57. C
58. C 2 CONTINUE
59. C
60. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
61. C
62. C UNITS(1)='UNITS'
63. C UNITS(2)='ERROR'
64. C PRINT 101, CODE
65. C 101 FORMAT(' ERROR--UNIT CODE', I3, ' NOT IN LINEAR TABLE.')
66. C RETURN
67. C END

```


FOR,5* N,LANGTME,,LANGTME

```

1* SUBROUTINE LANGTME(VALUE,UNICODE,UNITS)
2* C
3* C TABLE OF LANGTME MEASURE CONVERSION FACTORS.
4* C
5* C INTEGER ENTRY(3,2),UNITS(4),UNICODE
6* C
7* C FACTORS ARE DOUBLE PRECISION.
8* C
9* C DOUBLE PRECISION FACTOR(2)
10* C
11* C PUT UNIT CODES INTO ENTRY.
12* C
13* C DATA (ENTRY(1,1),1=1,2)/26,27/
14* C
15* C PUT IN ALPHA UNITS.
16* C
17* C DATA ((ENTRY(1,1),1=2,3),J=1,2)/'MONTHS',
18* C 'YEARS',
19* C
20* C ENTER INTERNAL UNITS.
21* C
22* C UNITS(3)=NO.
23* C UNITS(4)=
24* C
25* C IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
26* C
27* C IF (UNICDE.EQ.Q) UNICDE=-24
28* C
29* C SET CONVERSION FACTORS.
30* C
31* C DATA (FACTOR(1),1=1,2)/100,1200/
32* C
33* C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
34* C CODE INTO 'UNITS'.
35* C
36* C (CODE=TABLE(UNICDE)
37* C DO 2 1=1,2
38* C IF (ICCODE.NE.ENTRY(1,1)) GO TO 2
39* C IF (UNICDE.EQ.Q) VALUE=VALUE*FACTOR(1)
40* C IF (UNICDE.LT.Q) VALUE=VALUE/FACTOR(1)
41* C DO 1 J=1,2
42* C UNITS(J)=ENTRY(J+1,1)
43* C 1 CONTINUE
44* C
45* C AT THIS POINT, CONVERSION IS COMPLETE.
46* C
47* C RETURN
48* C
49* C ELSE, CHECK REST OF TABLE.
50* C
51* C 2 CONTINUE
52* C
53* C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
54* C

```

55. UNITS(1)=UNITS *
56. UNITS(2)=ERROR *
57. PRINT 101,ICODE
58. IUI FORMAT(1,ERROR--UNIT CODE,13,' NOT IN LONG TIME TABLE.')

59. RETURN
60. END

DELT,L N,MSQCVT,,MSQCVT

```

1. SUBROUTINE MSQCVT(MSQNO1,MSQNO2,LAT1,LON1,LAT2,LON2)
2.
3. PROGRAM TO CONVERT INPUT OF TWO MARSDEN SQUARES INTO
4. LATITUDES AND LONGITUDES OF TWO POINTS DEFINING THE
5. MINIMUM RECTANGLE INCLUDING THE MARSDEN SQUARES.
6.
7.
8. INITIALIZE THE FIRST SQUARE.
9.
10. DIMENSION LON(2),LAT(2)
11. 1 LNADD=10 0TO GET LONGITUDE.
12. 1 LSUB=0 0TO GET LATITUDE.
13. 1 LSGN=1 0SIGN FOR NORTHERN HEMISPHERE.
14. 1=1 0INDICATES FIRST SQUARE IS IN PROCESS.
15. MSQ=MSQNO1
16.
17. IF SQUARE IS IN SOUTHERN HEMISPHERE, BRANCH TO
18. CODE.
19.
20. 3 IF(MSQ.LT.300.OR.MSQ.GT.623) GO TO 4
21. MSQ=MSQ-264 0ADJUSTMENT FOR COMMON CODE BELOW.
22. 1 LSGN=-1 0SIGN FOR SOUTHERN HEMISPHERE.
23.
24. BRANCH TO CALCULATIONS.
25.
26. GO TO 5
27.
28. 4 IF(MSQ.GT.623) MSQ=MSQ-613
29. MSQ=MSQ-1
30.
31. CALCULATE LONGITUDE AND LATITUDE.
32.
33. 5 LON(1)=MOD(MSQ,36)*10*LNADD
34. LAT(1)=(MSQ/36*10*LNADD)
35.
36. ADJUST LONGITUDE TO ABS(LON(1)) *LE. 180.
37.
38. IF(LON(1).LT.180) LON(1)=LON(1)
39. IF(LON(1).GE.180) LON(1)=360-LON(1)
40.
41. IF DONE WITH THIS CASE, RETURN.
42.
43. IF(1.EQ.2) GO TO 99
44.
45. INITIALIZE FOR SECOND SQUARE.
46.
47. LNADD=0
48. LSUB=1
49. MSQ=MSQNO2
50. 1=2 0INDICATES SECOND SQUARE IN PROCESS.
51.
52. START SECOND PASS.
53.
54. GO TO 3

```

55. 99 LON1=LON(2)*1000
56. LON2=LON(1)*1000
57. LAT1=LAT(2)*1000
58. LAT2=LAT(1)*1000
59. RETURN
60. END

AD-A038 892

NAVAL SEA SYSTEMS COMMAND WASHINGTON D C

F/G 17/1

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAR - IN SUP--ETC(U)

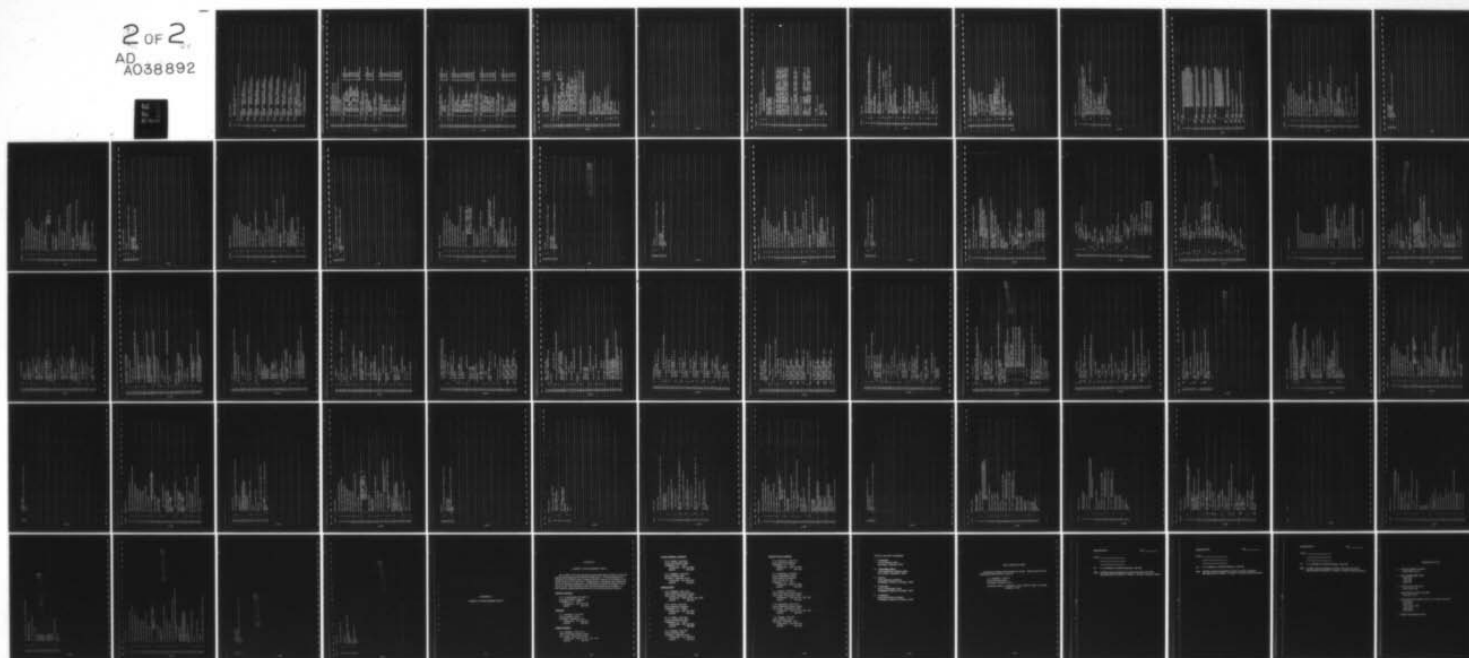
DEC 75

UNCLASSIFIED

NAVSEA-06H1/036-EVA/MOST-

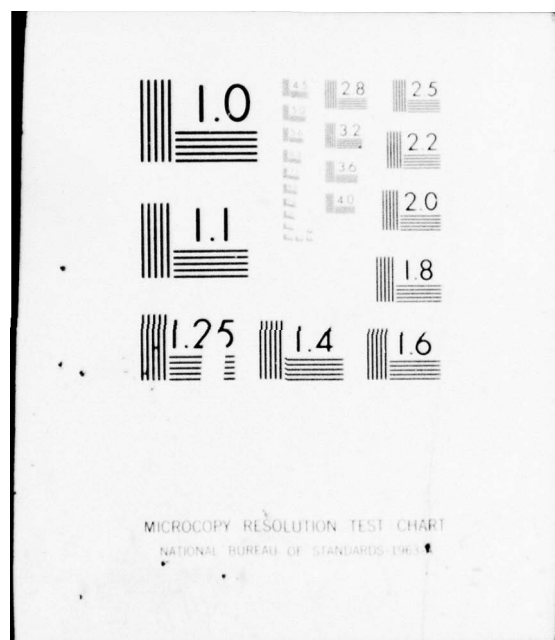
NL

2 OF 2
AD
A038892



END

DATE
FILMED
5-77



```

SUBROUTINE NAMEIT (ISET, ICODE, NAME, NR)
IMPLICIT INTEGER (A-Z)

```

.....

EXPERIMENT PARAMETERS (TABLE 16)

[illegible]

PARAMETER	10D16S=1, 10D16C=10D16S + NNT16 = 1
10D16S=1, 10D16C=10D16S + NNT16 = 1	

[illegible]

CLASS

PARAMETER Y175=1146.01, Y17E=1175.44

PARAMETER IWD17S=IWD16E+1,IWD17E=IWD17S+NA

.....

NAVIGATION PARAMETERS (TABLE 9)

PARAMETER Y9S=Y17E+1, Y9E=Y9S+26, NRT9=14

PARAMETER
INDS=IND17E+1,INDVE=IND9S+ART9-1-61N+56D+1,37101,1'[illegible]

WELSH SYSTEM

PARAMETER Y11S=Y9E+1,Y11E=Y11S+9,MRT

PARAMETER	INDIS=IND9E+,IND1E=IND1S+MRT
INDIS=IND9E+,IND1E=IND1S+MRT	INDIS=IND9E+,IND1E=IND1S+MRT

.....

Type 195

PARAMETER T125=7116+1.112E-1125+42, NHT12

PARAMETER
IWD12S=IWD11E+1, IWD12E=IWD12S+NR1

1. The first step in the process of identifying a problem is to recognize that a problem exists. This involves gathering information about the situation and identifying the specific issue that needs to be addressed.

DATA SOURCE PARAMETERS (TABLE 10)

PARAMETER $Y_{105} = Y_{126} + 1, Y_{106} = Y_{105} + 29, M8T10 = 9$

PARAMETER

USEY-7

ENVIRONMENTAL RISK PARAMETERS (TABLE 7)

PARAMETER $17S = Y10E + 1.17E - 17S + 36.8817 \div 14$

PARAMETER
WD7S=WD10E+1.0WD7E=WD7S+MMV7=1

THE UNIVERSITY OF CHICAGO

ACOUSTY

PARAMETER $T_{45}=17E+1, T_{4F}=Y_{45}+73, MRV_{45}=40$

PARAMETER 1404510D7E+1.10045=100450MRT9-1

[illegible]

DIMENSIONS

PARAMETER

```

DIMENSION IWORD(NMAX1),I0D(NMAX2),NAME(1)

```

DIMENSION INDEX(INGRP), INDEX(NGRP), NSETS(

DIMENSION IC (NGRP)

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
84

DATA (IOSTR(1),1,1,NGKP)/

140145.140175.140195.140115.140125.140165.140195/

55.	DATA (INDXST(1),1-1,NGRPT)/				
56.	1 T145,T175,T95,T115,T125,T105,T75,T45/				
57.	DATA (INDXND(1),1-1,NGRPT)/				
58.	1 T14E,T17E,T9E,T11E,T12E,T10E,T7E,T4E/				
59.	DATA (INSETS(1),1-1,NGRPT)/				
60.	1 NRT14,NRT17,NRT9,NRT11,NRT12,NRT10,NRT7,NRT4/				
61.	DATA (IC(1),1-1,NGRPT)/				
62.	1 0,0,1,0,1,1,0,0,0/				
63.	C				
64.	C.... EXPERIMENT NAMES (TABLE 14)				
65.	DATA (IMORD(1),1-T145,T14E)/				
66.	1 'FASOR 1 (REF: 024-003)'				
67.	2 'FASOR 11 (REF: 026-003)'				
68.	3 'FASOR 1 (REF: 042-001)'				
69.	4 'LORAD, HAWAII (REF: 008-001)'				
70.	5 'LORAD, ALASKA (REF: 198-003)'				
71.	6 'JOAST 111 (REF: 192-001)'				
72.	7 'JOAST 11 (REF: 192-002)'				
73.	8 'FASOR 111 (REF: 027-001)'				
74.	9 'GULF OF ALASKA (REF: 195-001)'				
75.	1 'SUOS 1 (REF: 005-003)'				
76.	0 /				
77.	C.... EXPERIMENT OFFSETS				
78.	DATA (IMD(1),1-IND145,IND14E)/				
79.	1 0,5,10,14,20,26,31,36,41,47/				
80.	C				
81.	C.... CLASSIFICATION NAMES (TABLE 17)				
82.	DATA (IMORD(1),1-T175,T17E)/				
83.	1 'UNCLASSIFIED'				
84.	2 'CONFIDENTIAL'				
85.	3 'SECRET'				
86.	0 /				
87.	C.... CLASSIFICATION OFFSETS				
88.	DATA (IMD(1),1-IND175,IND17E)/				
89.	1 0,2,4/				
90.	C				
91.	C.... NAVIGATION NAMES (TABLE 9)				
92.	DATA (IMORD(1),1-T95,T9E)/				
93.	1 'NOT AVAIL FOR APPL'				
94.	2 'SATELLITE'				
95.	3 'CELESTIAL'				
96.	4 'DEAD RECKON'				
97.	5 'LORAN C'				
98.	6 'EST. - SEE NOTES'				
99.	7 'LORAN A'				
100.	8 'DECCA'				
101.	9 'OMEGA'				
102.	1 'SMORAN'				
103.	1 'RAYDIST'				
104.	2 'COMB. - SEE NOTES'				
105.	3 'SINS'				
106.	4 'ACOUSTIC'				
107.	0 /				
108.	C.... NAVIGATION OFFSETS				
109.	DATA (IMD(1),1-IND95,IND9E)/				
110.	1 0,3,5,7,9,11,14,16,17,18,19,21,24,25/				
111.	C				

112.	C....	SYSTEM NAMES (SOURCE/RECV)	TABLE 11)	0	ISLT=4	0	ICDE,MND,OFFSET
113.		DATA (IMORD(1),I=1T15,TIME)/		0	1,2,0	0	
114.	1	*NOT APPLIC.		0	1,2,0	0	
115.	2	*AN/BQS-5		0	2,2,2	0	
116.	3	*AN/SQS-26 (AKR)		0	3,3,4	0	
117.	0			0		0	END SYS MMS
118.	C....	SYSTEM OFFSETS					
119.		DATA (IMORD(1),I=1T15,INDICE)/	0,3,1,1/				
120.	C						
121.	C....	TYPE NAMES (SOURCE/RECV)	TABLE 12)	0	ISLT=5	0	ICDE,MND,OFFSET
122.		DATA (IMORD(1),I=1T25,TIME)/		0	1,2,0	0	
123.	1	*NOT APPLIC.		0	1,2,0	0	
124.	2	*HORIZ. ARRAY		0	2,2,2	0	
125.	3	*OHNL. HYDROPHONE		0	3,3,4	0	
126.	4	*DIR. PROJ. - CM PULSED		0	4,4,7	0	
127.	5	*EXPLOSIVE		0	5,2,11	0	
128.	6	*AMBIENT NOISE		0	6,3,13	0	
129.	7	*SUS CHARGE, MK41		0	7,3,14	0	
130.	8	*DIR. HYDROPHONE		0	8,3,19	0	
131.	9	*OHNL. PROJ. - CM PULSED		0	9,3,22	0	
132.	1	*VERT. HYDROPHONE STRING		0	10,4,24	0	
133.	1	*VERT. ARRAY		0	11,3,30	0	
134.	2	*SUS CHARGE, MK02		0	12,3,32	0	
135.	3	*OHNL. PROJ. - FM PULSED		0	13,4,35	0	
136.	4	*DIR. PROJ. - FM PULSED		0	14,4,39	0	
137.	0			0		0	END TYP MMS
138.	C....	TYPE (SOURCE/RECV) OFFSETS					
139.		DATA (IMORD(1),I=1T125,INDICE)/					
140.	1	0,2,4,7,11,13,14,16,22,26,30,32,35,39/					
141.	C						
142.	C....	DATA SOURCE NAMES (TABLE 10)		0	ISLT=6	0	ICDE,MND,OFFSET
143.		DATA (IMORD(1),I=1T103,TIME)/		0	1,2,0	0	
144.	1	*NOT APPLIC.		0	1,2,0	0	
145.	2	*GRAPHICAL, PRELIMINARY		0	2,4,2	0	
146.	3	*GRAPHICAL, PUBLISHED		0	3,4,4	0	
147.	4	*TABULAR, PRELIMINARY		0	4,4,10	0	
148.	5	*TABULAR, PUBLISHED		0	5,4,14	0	
149.	6	*PUNCHED CARDS		0	6,3,18	0	
150.	7	*MAG. TAPE, DIGITAL		0	7,3,21	0	
151.	8	*MAG. TAPE, ANALOG		0	8,3,24	0	
152.	9	*PUNCHED PAPER TAPE		0	9,3,27	0	
153.	0			0		0	END DSR MMS
154.	C....	DATA SOURCE OFFSETS					
155.		DATA (IMORD(1),I=1T103,INDICE)/					
156.	1	0,2,4,10,14,18,21,24,27/					
157.	C						
158.	C....	ENVIR RUN NAMES (TABLE 7)		0	ISLT=7	0	ICDE,MND,OFFSET
159.		DATA (IMORD(1),I=1T51,TIME)/		0	1,1,0	0	
160.	1	*BT		0	2,1,1	0	
161.	2	*XBT		0	3,1,2	0	
162.	3	*XBT		0	4,1,3	0	
163.	4	*HYDROGRAPHIC CAST		0	5,4,6	0	
164.	5	*CORE, PISTON (GRAVITY)		0	6,2,10	0	
165.	6	*CORE, SHIPK		0	7,2,12	0	
166.	7	*GRAB SAMPLE		0	8,3,14	0	
167.	8	*THERMISTOR CHAIN		0	9,3,17	0	
168.	9	*DEPTH/SOUND VEL.		0		0	

```

169. 1 'SAL/TEMP/DEPTH' 0 10.3,20
170. 1 'SAL/TEMP/DEPTH/SV' 0 11.3,23
171. 2 'SEA SFC/IND SPEED TIME SERIES' 0 12.5,26
172. 3 'ENVIR. CONSTANTS ONLY' 0 13.9,31
173. 4 'BATHYMETRY' 0 14.2,35
174. 5 'GEND ENV MMS'
175. C
176. C... ENVIR OFFSETS
177. DATA (IND(1),1-10075,IND(7)/
178. 1 0,1,2,3,6,10,12,14,17,20,23,26,31,35/
179. C
180. C... ACOUSTIC RUN NAMES (TABLE 1) 0 ISET=0
181. DATA(IND(1),1-199,192)/
182. 1 'DIRECT PATH' 'SINGLE BR' 'M/U' 'SD' 'CZ'
183. 2 'SHALLOW WATER' 'BOT. REFL. LOSS - 1ST B.'
184. 3 'BOT. REFL. LOSS - NORMAL INC.' 'M/U'
185. 4 'VOLUME SCAT.' '2.0M/U'
186. 5 'AMBIENT NOISE' '2.0M/U'
187. 6 'BOT. REFL. LOSS - 2ND B.' '2ND BR'
188. 7 'SD, BR, CZ' 'DEEP REFR.' 'SURFACE SCAT.'
189. 8 'SD, ASR' 'SD, DEEP REFR.' '3RD BR'
190. 1 'BOT. REFL. LOSS - 3RD B.' 'SD, DP' 'BR CZ'
191. 2 'SD, BR' 'ISOFR' 'M/U' 'BEAN PATTERN'
192. 3 '2.0M/U' 'R.A.P.' 'SD, BC' 'M/U'
193. 4 'SHADOW ZONE' 'MULTI - MODE' 'VERT. BEAN PAT. - CZ'
194. 5 'VERT. BEAN PAT. - BB/00T' 'ISOFR TO SD'
195. C
196. C... ACOUSTIC RUN OFFSETS (LT 0 NOT PRECEDED WITH PROP. LOSS 1)
197. DATA(IND(1),1-10045,10041/
198. 1 0,2,-4,5,6,7,-10,-14,-19,-20,-22,-23,-24,-27,-28,-29,
199. 2 33,34,36,-30,40,41,43,-44,48,49,50,51,-52,-53,-55,-56,
200. 3 57,58,-59,60,62,-64,-68,72/
201. C
202. C
203. NR=0
204. ICDE=ICDE+IC(ISET)
205. JMP=1
206. IF(ICDE.LE.0) GO TO 999
207. IF(INSET(ISET))-ICDE,999,50,60
208. JMP=2
209. IOFF=IOFFSTR(ISET)+ICDE-1
210. IF(ISET.NE.0) GO TO 65
211. IF(IND(IOFF).LT.0) GO TO 65
212. NR=2
213. NAME(1)=PROP.
214. NAME(2)=LOSS
215. 45 1ST=INDXSTR(ISET)+ABS(IND(IOFF))
216. GO TO 170,60,JMP
217. 70 IEND=INDXSTR(ISET)+ABS(IND(IOFF+1))-1
218. GO TO 90
219. 80 IEND=INDXSTR(ISET)
220. 90 DO 100 I=1ST,IEND
221. KR=NR+1
222. 100 NAME(NR)=IND(1)
223. RETURN
224. 999 NR=1
225. NAME(1)=.....

```

226. RETURN
227. END

FOR S4 N-NAMGET, NAMGET

```

1. SUBROUTINE NAMGET (CODE, NAME)
2. C
3. C NAMGET (NAME GET) RETURNS THE ALPHA NAME IN 'NAME' FOR
4. C THE NAME CODE 'CODE' ENTERED.
5. C
6. C IMPLICIT INTEGER (A-Z)
7. C REAL*8 DATA(149), TEMP
8. C DIMENSION N(2), NAME(2)
9. C EQUIVALENCE (TEMP, N(1))
10. C
11. C DATA (DATA(1), I=1, 72) /
12. C
13. C 'SOURCE DEPTH', 'RECVR DEPTH', 'FREQUENCY', 'FREQ. BAND', '
14. C 'SOURCE LEVEL', 'PULSE LENGTH', 'WATER DEPTH', 'TRANS. DEPTH',
15. C 'COLUMN SRTM', 'SMALLER D.L.', 'GREATER D.L.', 'PROP. LOSS', '
16. C 'HORIZ. RANGE', 'RTH REF LOSS', 'GRAZING ANGLE', 'LOB FREQ LTM',
17. C 'H1 FREQ LTM', 'VOL SCAT STR', 'DEPTH', 'TRAVEL TIME', '
18. C 'SUR SCAT STR', 'RPTH SCE DTH', 'RPTH RCR DTH', 'TARGET STR', '
19. C 'REVERB. LVL', 'REPT. RATE', 'REL RESPONSE', 'ANGULAR DEV.', '
20. C 'GEO MEAN FRQ', 'DEPRESS. ANG', 'HOR REPEATER', 'HNGE/DPTH SS',
21. C 'S HOR BEAM', 'S VERT BEAM', 'M HOR BEAM', 'M VERT BEAM',
22. C '360 NOT USED' /
23. C
24. C DATA (DATA(1), I=73, 85) /
25. C
26. C 'WAVE DIR COO', 'WAVE HEIGHT', 'WAVE PERIOD', 'SEA STATE', '
27. C 'WIND DIR', 'WIND SPEED', 'SWELL DIR CO', 'SWELL HEIGHT',
28. C 'SWELL PERIOD', 'BEATHEN CODE', 'BOTTOM SLOPE', 'NOT USED', '
29. C 'BOTTOM DEPTH' /
30. C
31. C DATA (DATA(1), I=86, 149) /
32. C
33. C 'DEPTH', 'NOT USED', 'TEMPERATURE', 'SALINITY', '
34. C 'SOUND SPEED', 'SIGNA-T', 'H1 DEPTH LTM', 'LO DEPTH LTM',
35. C 'DENSITY', 'POROSITY', 'SAND', 'SILT',
36. C 'CLAY', 'GRAIN SIZE', 'TRK DISTANCE', 'TIME OF DAY', '
37. C 'WAVE DIR', 'SWELL DIR', 'HORIZ. RANGE', 'ATTEN FACTOR',
38. C 'SS GRADIENT', 'THICKNESS', '37 NOT USED' /
39. C
40. C
41. C
42. C J = CODE
43. C IF (J.GT. 72) J = J - 28
44. C
45. C TEMP = DATA(I)
46. C NAME(1) = N(1)
47. C NAME(2) = N(2)
48. C
49. C RETURN
50. C END

```


DELT.L N.NOTE..NOTE

```

1: SUBROUTINE NOTE (DATA)
2: C
3: C ROUTINE FOR THE SELECTIVE PRINTING OF NOTES. THE USER RESPONDS TO
4: C QUESTIONS ASKING THE EXPERIMENT NUMBER AND CRUISE NUMBER FOR WHICH NOTES
5: C ARE TO BE PRINTED.
6: C
7: C IMPLICIT INTEGER (A-Z)
8: C DIMENSION DATA(1),NAME(10)
9: C
10: WRITE (6,5)
11: 5 FORMAT (1: SPECIFY THE EXP. AND CRUISE NUMBER FOR WHICH NOTES ARE Y
12: 10 BE PRINTED)
13: C
14: 10 PHATE = 242195 B OCTAL 00001000001
15: C
16: C INPUT EXPERIMENT NUMBER AND CRUISE NUMBER FOR NOTES
17: C
18: 15 WRITE (6,20)
19: 20 FORMAT (1: ENTER EXPERIMENT NO.: CRUISE NO. / ENTER '.5H.END.')
20: READ (5,30,ERR=90) FLAG
21: 30 FORMAT (11)
22: IF (FLAG.EQ.'E') GO TO 900
23: READ (10,35,ERR=90) EXPNO,CRUSNO
24: 35 FORMAT (1)
25: GO TO 50
26: C
27: C ERROR
28: C
29: 40 WRITE (6,45)
30: 45 FORMAT (1: ... READ FORMAT ERROR ....)
31: GO TO 15
32: C
33: 50 FLD(10,10,SECADR) = FLD(10,10,PHATE)
34: FLD (10,10,MSEC) = FLD(10,10,PHATE)
35: C
36: CALL DSRED (SECADR,MSEC,DATA)
37: C
38: EXNO = DATA(2)
39: PHATE = DATA(15)
40: PCRU = DATA(16)
41: C
42: C CHECK IF REQUESTED EXPERIMENT
43: C
44: IF (EXNO.EQ.EXPNO) GO TO 40
45: C
46: C CHECK FOR NEXT EXPERIMENT
47: C
48: 55 IF (PHATE) 50,50
49: C
50: WRITE (6,57)
51: 57 FORMAT (1: ... SPECIFIED RECORD NOT FOUND ....)
52: GO TO 10
53: C
54: 60 PHATE = PCRU

```

```

55. C 70 FLD(10,10,SECADR) = FLD(10,10,PHATC)
56. FLD(10,10,SEC) = FLD(10,10,PHATC)
57. C
58. C
59. CALL OSKRED (SECADR,SEC,DATA)
60. CRUNO = FLD(10,10,DATA(2))
61. PHATC = DATA(3)
62. C
63. C CHECK IF REQUESTED CRUISE
64. C
65. C IF (CRUNO.EQ.CRUSNO) GO TO 80
66. C
67. C IF (PHATC) 70,55,70
68. C
69. C PRINT NOTES
70. C
71. 80 NCARDS = FLD(10,10,DATA(2))
72. IF (NCARDS.EQ.0) GO TO 100
73. N = 14*NCARDS + 17
74. CALL NAMEIT (1,EXNO,NAME,NR)
75. WRITE (6,90) EXNO,NAME(1),1-1,NR
76. FORMAT (1M), 'NOTES FOR EXPERIMENT',15,2,10A4)
77. WRITE (6,91) CRUNO,10DATA(1),1-10,N
78. 91 FORMAT (1I3,'CRUISE',15//11X,13A6,A2))
79. GO TO 10
80. WRITE (6,110) EXNO,CRUNO
81. 110 FORMAT ('THERE ARE NO NOTES FOR EXPERIMENT',15/
82. 1725,'CRUISE',15)
83. GO TO 10
84. C
85. 900 WRITE (6,910)
86. 910 FORMAT (3X,'END')
87. RETURN
88. END

```

OFOR,SW H.PAKATB,PAKATB

```

1. SUBROUTINE PAKATB (ITEMS,DSITEMS,MITEMS,ITABLE)
2.
3. C PROGRAM TO PRINT THE NUMBER OF RUNS FOR CONSTANTS, AND THE NUMBER
4. C OF RUNS AND DATA SETS FOR PARAMETERS AND VARIABLES (IF NUMBER).
5. C (ITABLE=2 FOR ACOUSTIC, ITABLE=5 FOR ENVIRONMENTAL).
6. C IMPLICIT INTEGER (A-Z)
7. DIMENSION ITEMS(1),DSITEMS(1),NAME(2)
8. WRITE (6,10)
9. FORMAT (120,'NUMBER',T40,'NUMBER OF',/ CODE',T12,'PARAMETER NAME
10. 'T30,'OF RUNS',T40,'DATA SETS')
11. CODE = 0
12. IF (ITABLE.EQ.5) CODE = 100
13. DO 30 I = 1,MITEMS
14. CODE = CODE + 1
15. IF (ITEMS(I).EQ.0) GO TO 30
16. CALL NAMEGET (CODE,NAME)
17. ENCODE (6,15,VALUE) DSITEMS(I)
18. IF (DSITEMS(I).EQ.0) VALUE = ' '
19. WRITE (6,20) CODE,NAME(1),NAME(2),ITEMS(I),VALUE
20. FORMAT (14,13) - '2A6,T29,16,T40,A6)
21. CONTINUE
22. RETURN
23. END
24.

```



```

1. RETREV PROC
2. COMMON /RETREV/NLAT,ELON,SLAT,MLON,PRV,TOYR,MCON,NORTHST(12),
3. FDAY,TODAY,ENTIM,TOTIM,NOSEX,SEX(10),NACDCR,
4. ACDCSTDTJ,NACPAR,ACPAR(14,10),MENDCN,MENDCNST(10),
5. MNPAC,CNPAR(14,10),RUNTYP,MPTRS,MSTYPE,STPE(10),
6. MTYPE,MTYPE(10),MSSYST,SSYST(10),MRSYST,MRSYST(10),
7. NANTHD(10),ANTHD(10,10),MANTHD(10),ENTHD(10,10),
8. ANV,EXP,CNU,STA,RON,CESREQ,YRGR,TRAKEST,WASK(3),
9. RQUEST(3),DAY,DAY,YEAR,TIME,MINUTE,SECOND
10. END
11. EXPNAT PROC
12. COMMON /EXPNAT/ERNO,PHATE,PCRU,EXLAT,EXLON,EXDAT,EXFLAT,EXFLON,
13. EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
14. EASTAT(2),EEESTAT(2)
15. END
16. CRUISE PROC
17. COMMON /CRUISE/CRUNO,PHATC,CALAT,CALON,CAIDAT,CAFLAT,CAFLON,
18. CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
19. CASTAT(2),CESTAT(2),PACS,PENS,MCARDS,NOTES(14,132)
20. END
21. ASTATN PROC
22. COMMON /ASTATN/SANO,SAMON,SAILAT,SALON,SAIDAT,SAFLAT,SAFLON,
23. SAFDAT,SASTAT(2),PHATAS,PENUS
24. END
25. RUN PROC
26. COMMON /RUH /PORA,RILAT,RINS,REFAT,RPNS,RILON,RIEN,RFLOW,RFEB,
27. RIHO,RIIDAT,RIYR,RIYR,RIZON,RIYR,MFDAY,RIYR,MFTIM,
28. RPZCN,NAVCD,DTASRC,SYSSRC,SYSSRC,SRCCTP,RCRTYP,
29. CLAS,DESCR,NAVDIR,NAVMT,NTUNIT,KAYPRD,PRDUNT,
30. SEASTA,ENDOTR,UNVEL,VELUNIT,SWLDIR,SWLMT,
31. SMTUNT,SMLPRO,SPDUNT,WEATHR,BOTDPH,DPMUNT,BTMSLP,
32. BATHY,INFO(4),RSTAT(2),RUNNO
33. END
34. CPU PROC
35. COMMON /CPU /NCON,CON(9,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
36. NDAYS,NHONST(50),NDATA,VALNG(2,30),50,DATA(3000)
37. END
38. SECTO PROC
39. COMMON /SECTO /ID(7),IDATE,DUNNY(20)
40. END
41. DATCOM PROC
42. COMMON /DATCOM/NDUSED,NRS(50)
43. END
44. GETLST PROC
45. COMMON /GETLST/EXPI,CRUI,STAI,RUNI,ALL,MNUNSF,USERAP,AETYPE
46. LOGICAL ALL,USERAP
47. END

```


1.	C	SUBROUTINE PERCENTIVALUE,UNICODE,UNITS,
2.	C	
3.	C	TABLE OF PERCENT MEASURE CONVERSION FACTORS.
4.	C	
5.	C	INTEGER ENTRY(1),UNITS(1),UNICODE
6.	C	
7.	C	FACTORS ARE DOUBLE PRECISION.
8.	C	
9.	C	DOUBLE PRECISION FACTOR(I)
10.	C	
11.	C	PUT UNIT CODES INTO ENTRY.
12.	C	
13.	C	DATA (ENTRY(1),1)=1,31/42,43,44/
14.	C	
15.	C	PUT IN ALPHA UNITS.
16.	C	
17.	C	DATA (ENTRY(1),1)=2,31/42,43,44//PER SEC,MNT
18.	C	'(0/00)','','PPH ''
19.	C	
20.	C	ENTER INTERNAL UNITS.
21.	C	
22.	C	UNITS(I)='PERCENT'
23.	C	UNITS(I)='MNT'
24.	C	
25.	C	IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
26.	C	
27.	C	IF (UNICODE.EQ.0) UNICODE=-42
28.	C	
29.	C	SET CONVERSION FACTORS.
30.	C	
31.	C	DATA (FACTOR(1),1)=1,31/1.00,0.100,1.00-4/
32.	C	
33.	C	IF CODE IS IN TABLE, PERFORM CONVERSION; ALSO, ENTER
34.	C	CODE INTO 'UNITS'.
35.	C	
36.	C	ICODE=LABS(UNICODE)
37.	C	DO 2 I=1,3
38.	C	IF ICODE.NE.ENTRY(I,1) GO TO 2
39.	C	IF (UNICODE.EQ.0) VALUE=VALUE*FACTOR(I)
40.	C	IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
41.	C	DO 1 J=1,2
42.	C	UNITS(J)=ENTRY(J+1,1)
43.	C	I CONTINUE
44.	C	
45.	C	AT THIS POINT, CONVERSION IS COMPLETE.
46.	C	
47.	C	RETURN
48.	C	
49.	C	ELSE, CHECK REST OF TABLE.
50.	C	
51.	C	CONTINUE
52.	C	
53.	C	IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNIT
54.	C	

```

55.  UNITS(1)=UNITS 1
56.  UNITS(2)=ERROR 1
57.  PRINT 101,ICODE
58.  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PER CENT TABLE.')
59.  RETURN
60.  END

```

DFOR,SM N,PLANAR,,PLANAR

```

1. SUBROUTINE PLANAR(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF AREA MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,4),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(4)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1=1,4)/(3,14,15,16/
14. C
15. C PUT IN ALPHA UNITS.
16. C DATA ((ENTRY(1,J),J=1,2),J=1,4)/('CM',2,1,1
17. C 'METERS',1,002,1,1
18. C 'FEET',1,12,1,1
19. C 'YARDS',1,1,2,1,1
20. C
21. C ENTER INTERNAL UNITS.
22. C
23. C UNITS(3)=METERS
24. C UNITS(4)=SQ.
25. C
26. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
27. C
28. C IF(UNICODE.EQ.0) UNCODE=-14
29. C
30. C SET CONVERSION FACTORS.
31. C
32. C DATA (FACTOR(1),1=1,4)/(10000.000,1.000,7.624900,.836100/
33. C
34. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
35. C
36. C UNITS(1)='UNITS'
37. C UNITS(2)='ERROR'
38. C
39. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
40. C CODE INTO 'UNITS'.
41. C
42. C ICODE=IABS(UNICODE)
43. C DO 2 I=1,4
44. C IF(ICODE.NE.ENTRY(1,I)) GO TO 2
45. C IF(UNICODE.GE.0) VALUE=VALUE*FACTOR(I)
46. C IF(UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
47. C DO 1 J=1,2
48. C UNITS(J)=ENTRY(J,1)
49. C 1 CONTINUE
50. C
51. C AT THIS POINT, CONVERSION IS COMPLETE.
52. C
53. C RETURN
54. C

```



```

55. C ELSE, CHECK REST OF TABLE.
56. C
57. C 2 CONTINUE
58. C
59. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
60. C
61. C UNITS(1)='UNITS '
62. C UNITS(2)='ERROR '
63. C PRINT 101,ICODE
64. C I/O FORMAT(1) ERROR--UNIT CODE(1,2), NOT IN AREA TABLE.
65. C RETURN
66. C END

```


Н. ПОМОН, . ПОМОН

```

1. SUBROUTINE POWDER(VALUE,UNICODE,UNITS)
2.
3. C TABLE OF POWER MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTNT(3,2),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(2)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTNT(1,1),1,1,2)/59,60/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTNT(1,1),1,2,3),J=1,2)/'ERGS/S','EC. ',
18. C 'WATTS ','/'
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(3)='WATTS/'
23. C UNITS(4)=' '
24. C
25. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF(UNICODE.EQ.0) UNCODE=-40
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA (FACTOR(1),1,1,2)/1.00-7,100/
32. C
33. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
34. C
35. C UNITS(1)='UNITS '
36. C UNITS(2)='ERROR '
37. C
38. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS' ALPHA
39. C CODE INTO 'UNITS'.
40. C
41. C ICODE=IAUS(UNICODE)
42. C DO 2 1,2
43. C IF(ICODE.NE.ENTNT(1,1)) GO TO 2
44. C IF(UNCODE.GE.0) VALUE=VALUE*FACTOR(1)
45. C IF(UNCODE.LT.0) VALUE=VALUE/FACTOR(1)
46. C UNITS(1)=ENTNT(2,1)
47. C UNITS(2)=ENTNT(3,1)
48. C
49. C AT THIS POINT, CONVERSION IS COMPLETE.
50. C
51. C RETURN
52. C
53. C ELSE, CHECK REST OF TABLE.
54. C

```

2 CONTINUE

C

IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.

C

C

PRINT I01,ICODE

I01 FORMAT: 'ERROR--UNIT CODE',I3,' NOT IN POWER TABLE.')

RETURN

END

BF0R,5# N,PRESSR,,PRESSR

```

1. SUBROUTINE PRESSR(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF PRESS MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(1,9),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(9)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1,1,1,9),40,49,50,51,52,53,54,55,56/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA (ENTRY(1,2),1,2,3),J=1,9)/'MICRO ','BARS ','
18. C 'PASCAL','PS','MICRO ','IPAS
19. C 'KG./CM','...2 ','LBS./I','IN...2 ','
20. C 'LBS./FT','T...2 ','IN. ME','MERCURY ','
21. C 'MM. HG.','MERCURY ','ATMOSP.','MERES '/'
22. C
23. C ENTER INTERNAL UNITS.
24. C
25. C UNITS(1)=INHG/CM*1
26. C UNITS(4)=*2
27. C
28. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
29. C
30. C IF (UNICODE.EQ.0) UNICODE=-51
31. C
32. C SET CONVERSION FACTORS.
33. C
34. C DATA (FACTOR(1,1),1,1,9)/101,97100,1.020-3,1.0200,100,.703100,
35. C 4.0820-3,.345362700,0.772212500,
36. C 1.03322280706/
37. C
38. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
39. C CODE INTO 'UNITS'.
40. C
41. C ICODE=IABS(UNICODE)
42. C DO 2 1=1,9
43. C IF (ICODE.NE.ENTRY(1,1)) GO TO 2
44. C IF (UNICODE.GE.0) VALUE=VALUE*FACTOR(1)
45. C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(1)
46. C DO 1 J=1,2
47. C UNITS(J)=ENTRY(1,J+1,1)
48. C 1 CONTINUE
49. C
50. C AT THIS POINT, CONVERSION IS COMPLETE.
51. C
52. C RETURN
53. C
54. C ELSE, CHECK REST OF TABLE.

```



```

55. C
56. 2 CONTINUE
57. C
58. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
59. C
60. UNITS(1)='UNITS'
61. UNITS(2)='ERROR'
62. PRINT IOL, CODE
63. 101 FORMAT(' ERROR--UNIT CODE', I3, ' NOT IN PRESSURE TABLE.')
```

```

RETURN
END
```

BFOR,SA N,PSLV,PSLV

```

1. SUBROUTINE PSLV(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF PRESSURE LEVEL MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(3,3),UNITS(4),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(3)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(1,1),1,1,3)/57,70,71/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA ((ENTRY(1,1),1,1,3),J=1,3)/DB//IU,IBAR
18. C
19. C ENTER INTERNAL UNITS.
20. C
21. C
22. C UNITS(3)=DB//IU
23. C UNITS(4)=PAS
24. C
25. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
26. C
27. C IF(UNICODE.EQ.0) UNICDE=71
28. C
29. C SET CONVERSION FACTORS.
30. C
31. C DATA (FACTOR(1,1),1,1,3)/-100.000,-26.000,000/
32. C
33. C INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
34. C
35. C UNITS(1)=UNITS
36. C UNITS(2)=ERROR
37. C
38. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
39. C CODE INTO 'UNITS'.
40. C
41. C ICODE=IABS(UNICODE)
42. C DO 2 I=1,3
43. C IF(ICODE.NE.ENTRY(1,1)) GO TO 2
44. C IF(UNICODE.GE.0) VALUE=VALUE*FACTOR(I)
45. C IF(UNICODE.LT.0) VALUE=VALUE*-FACTOR(I)
46. C UNITS(1)=ENTRY(2,1)
47. C UNITS(2)=ENTRY(3,1)
48. C
49. C AT THIS POINT, CONVERSION IS COMPLETE.
50. C
51. C RETURN
52. C
53. C ELSE, CHECK REST OF TABLE.
54. C

```

```

55.      2 CONTINUE
56.      C
57.      C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
58.      C
59.      PRINT 101,ICODE
60.      101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PRESSURE LEVEL TABLE.')
```

```

61.      RETURN
62.      END
```

QELV,L N,RANGE,,RANGE

1. SUBROUTINE RANGE (MTHNGS,THNGS,,NMTHDS,MTHDS,MTHNGS)

2. C

3. IMPLICIT INTEGER (A-Z)

4. C

5. C

6. COMMON /CPV 7NCON,COMIT,30,NPAR,PART30,301,NVAR,IVAR(3,33),

7. NDATS,NHONS(50),NDATA,VALRNG(2,30,50),DATA(30000)

8. DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)

9. EQUIVALENCE (ICON(1,1),ICON(1,1)),(IPAR(1,1),IPAR(1,1)),

10. (VALRNG(1,1),IVLRNG(1,1)),(IVLRNG(1,1),IVLRNG(1,1))

11. REAL COM,PA,DATA,VALRNG

12. C

13. C

14. DIMENSION THNGS(4,10),UNITS(4),PSUB(30),NMTHDS(10),MTHDS(10,10)

15. DIMENSION ATHNGS(4,10),ITHNGS(4,10),RTHNGS(4,10)

16. REAL ATHNGS,RTHNGS

17. EQUIVALENCE (ATHNGS,ITHNGS), (ITHNGS,RTHNGS)

18. C

19. COMMON /DATCON/ NDUSED,NRS(50)

20. C

21. IF (MTHNGS.NE.0) GO TO 3

22. NDUSED=NDATS

23. DO 2 I=1,NDATS

24. NRS(I)=1

25. 2 CONTINUE

26. RETURN

27. C

28. C LOOP TO MOVE THNGS TO ATHNGS

29. C

30. 3 DO 10 I=1,MTHNGS

31. DO 5 J=1,4

32. ITHNGS(J,I)=THNGS(J,I)

33. 5 CONTINUE

34. 5 CALL CONVERT (ATHNGS(2,1),ITHNGS(1,1),ITHNGS(4,1),

35. UNITS)

36. CALL CONVERT (ATHNGS(3,1),ITHNGS(1,1),ITHNGS(4,1),

37. UNITS)

38. 10 CONTINUE

39. C

40. IF (NCON.EQ.0) GO TO 126

41. C

42. DO 100 J=1,MTHNGS

43. NM=NMTMDS(J)

44. C

45. DO 90 I=1,NCON

46. C

47. IF (ICON(1,I).NE.THNGS(1,J)) GO TO 90

48. C

49. IF (RTHNGS(2,J).EQ.-99999999.) GO TO 50

50. IF (CON(3,I).LT.ATHNGS(2,J)) RETURN 3

51. C

52. IF (RTHNGS(3,J).EQ.-99999999.) GO TO 60

53. IF (CON(3,I).GT.ATHNGS(3,J)) RETURN 3

54. C


```

55. C CHECK METHODS
56. C
57. 40 IF (NM.EQ.0) GO TO 90
58. DO 70 L = 1, NM
59. IF (INTD5(L, J).EQ.ICON(4, 1)) GO TO 90
60. CONTINUE
61. C
62. 70 METHODS DID NOT MATCH
63. RETURN 3
64. C
65. 90 CONTINUE
66. C
67. 100 CONTINUE
68. C
69. C INITIALIZE MDOUSED
70. C
71. 125 MDOUSED = 0
72. C
73. DO 130 I = 1, 30
74. PSUB(I) = 1
75. 130 CONTINUE
76. C
77. C
78. C IF (MDATS.EQ.0) RETURN
79. C
80. DO 500 J = 1, MDATS
81. C
82. 01. C
83. IF (I.EQ.1) GO TO 150
84. C
85. J = NPAR
86. PSUB(J) = PSUB(J) + 1
87. IF (PSUB(J).LE.1PAR(1, J)) GO TO 150
88. PSUB(J) = 1
89. J = J + 1
90. GO TO 140
91. C
92. C LOOP FOR REQUESTS
93. C
94. 150 DO 150 K = 1, NTHNGS
95. NM = NTHNGS(K)
96. C
97. C CHECK FOR PARAMETERS
98. C
99. IF (NPAR.EQ.0) GO TO 300
100. C
101. C LOOP FOR PARAMETERS
102. C
103. DO 290 J = 1, NPAR
104. C
105. IF (1PAR(2, J).NE.1THNGS(1, K)) GO TO 290
106. C
107. JJ = PSUB(J) + 4
108. IF (1THNGS(2, K).EQ.-99999999) GO TO 200
109. IF (1PAR(JJ, J).LE.1THNGS(2, K)) GO TO 500
110. C
111. IF (1THNGS(3, K).EQ.-99999999) GO TO 210
112. IF (1PAR(JJ, J).LE.1THNGS(3, K)) GO TO 500

```

```

112. C
113. C
114. C
115. 210
116. IF (NM.EQ.C) GO TO 293
117. DO 220 L = 1,NM
118. IF (INTD5(L,K).EQ.IPAR(4,J)) GO TO 290
119. CO,INUE
120. C
121. METHODS DID NOT MATCH
122. GO TO 530
123. C
124. 290
125. C
126. C
127. 300
128. C
129. C
130. C
131. C
132. C
133. C
134. 350
135. C
136. C
137. C
138. C
139. 360
140. C
141. C
142. 370
143. C
144. C
145. C
146. C
147. C
148. C
149. C
150. C
151. C
152. C
153. C
154. C
155. C
156. C
157. C
158. C
159. C
160. C
161. C

```

CHECK METHODS

IF (NM.EQ.C) GO TO 293
DO 220 L = 1,NM
IF (INTD5(L,K).EQ.IPAR(4,J)) GO TO 290
CO,INUE

METHODS DID NOT MATCH
GO TO 530

CONTINUE

LOOP FOR VARIABLES

DO 100 J=1,NVAR

IF (IPAR(1,J).NE. THNGS(1,K)) GO TO 400

IF (ITHNGS(2,K).EQ. -99999999.) GO TO 450

IF (IVALRNG(2,J)) .LT. ATHNGS(2,K)) GO TO 500

IF (ITHNGS(3,K).EQ. -99999999.) GO TO 360

IF (IVALRNG(1,J)) .GT. ATHNGS(3,K)) GO TO 500

CHECK METHODS

IF (NM.EQ.C) GO TO 400

DO 370 L = 1,NM

IF (INTD5(L,K).EQ.IPAR(3,J)) GO TO 400

CONTINUE

METHODS DID NOT MATCH
GO TO 500

CONTINUE

CONTINUE

INCREMENT COUNTER

NDUSED = NDUSED + 1
NRS(NDUSED) = 1

CONTINUE

IF (NDUSED .EQ. C) RETURN 3
RETURN

END

BEST AVAILABLE COPY

```

DELTA      N=SELECT,=SELECT

1:         C
2:         C
3:         C
4:         C
5:         C
6:         C
7:         C
8:         C
9:         C
10:        C
11:        C
12:        C
13:        C
14:        C
15:        C
16:        C
17:        C
18:        C
19:        C
20:        C
21:        C
22:        C
23:        C
24:        C
25:        C
26:        C
27:        C
28:        C
29:        C
30:        C
31:        C
32:        C
33:        C
34:        C
35:        C
36:        C
37:        C
38:        C
39:        C
40:        C
41:        C
42:        C
43:        C
44:        C
45:        C
46:        C
47:        C
48:        C
49:        C
50:        C
51:        C
52:        C
53:        C
54:        C

PARAMETER NACPT=36, NEMPT=35, NACDT=30, NENDT=19
IMPLICIT INTEGER (A-Z)
LOGICAL LISTF, FIRST
INCLUDE RETREV, LIST
INCLUDE EPHNT, LIST
INCLUDE CRUISE, LIST
INCLUDE ASTATN, LIST
INCLUDE RUN, LIST
INCLUDE SPV, LIST
INCLUDE SECT, LIST
INCLUDE DATCOM, LIST
REAL NAVDIR, NAVMT, NAVPRD, SEASTA, ANDDIR, ENDVEL, SMLDIR,
  SMLMT, SMLPRD, SEATHR, SOTDPH, BTNSLP
DIMENSION ICON(4,30), IPAR(30,30), IVALRG(2,30,50)
EQUIVALENCE (CON(1,1), ICON(1,1)), (PAR(1,1), IPAR(1,1)),
  (VALRG(1,1), IVALRG(1,1)), (AEQUIV, ACPARS),
  (EEQUIV, ENPARS)
REAL CON, PAR, DATA, VALRG
REAL AEQUIV(4,10), EEQUIV(4,10), AORD
DIMENSION IUT(1:56), DISK(7), ASTATS(2), ESTATS(2)
DIMENSION IOR(3000), CHAN(81)
DATA DISK / 1 NAVSEA UNDERWATER ACOUSTIC DATA BANK 1/
DIMENSION ACPT(NACPT), ENPT(NEMPT), ACOT(NACDT), ENDT(NENDT),
  IACPOST(NACPT), ENPOST(NEMPT), IAPUT(NENDT), IMPPT(NEMPT),
  IMPUST(NEMPT)
LISTF = .FALSE.
FIRST = .TRUE.
INIT. UNIT 13 50 WE CAN USE SETADR WITH NTRAN

```

BEST AVAILABLE COPY

```

55. C CALL NTRAN(10,22)
56. C
57. C HEAD IN SECTION ZERO
58. C
59. C CALL DSPRED (Q,I,10)
60. C
61. C MAKE SURE IT'S CONNECT DISK
62. C
63. C
64. C DO IC I=1,7
65. C IF (DISK(I) .NE. 10(I)) GO TO 15
66. C IO CONTINUE
67. C GO TO 18
68. C
69. C WRITE ERROR
70. C
71. C IS WRITE(6,16)
72. C 14 FORMAT (' ERROR -- WRONG DISK MOUNTED')
73. C STOP
74. C
75. C 18 CALL ENTRAN ('DATES',DATE,TIME)
76. C FLD(C,12,DATE) = FLD(12,12,DATE)
77. C FLD(C,12,PLAN) = FLD(24,12,DATE)
78. C FLD(C,12,MINUTE) = FLD(12,12,TIME)
79. C FLD(C,12,SECOND) = FLD(24,12,TIME)
80. C IF (FIRST) WRITE (6,25) (ID(I),I=1,7),DATE,DATE,TIME,MINUTE,
81. C SECOND
82. C 25 FORMAT ('1',T30,'NAYDAB',T12,'A6,A3,')T14,11/T23,21A2,
83. C 17',12,4X,2(A2,'11,A2//1X,'TO RETURN TO PREVIOUS QUESTION. ENTE
84. C 14 CLUF')
85. C 20 PNATE = 262145 W OCTAL 0000100001
86. C
87. C READ RETRIEVAL DATA
88. C
89. C 26 CALL INPUT (FIRST)
90. C FIRST = .FALSE.
91. C
92. C CALL STORE
93. C
94. C INITIALIZE INVENTORY COUNTERS
95. C
96. C DO 31 I = 1,NACPT
97. C ACPOST(I) = 0
98. C
99. C 31 ACPT(I) = 0
100. C DO 32 I = 1,NENPT
101. C ENPOST(I) = 0
102. C ENPT(I) = 0
103. C DO 33 I = 1,NACDT
104. C ACOT(I) = 0
105. C DO 34 I = 1,NENDT
106. C ENDT(I) = 0
107. C
108. C DATYPE=0 W SET ACOUSTIC FLAG
109. C
110. C IF (RUNTYP.EQ.2) DATYPE = 1
111. C

```



```

112. C      INIT NCONTA, NCONTE
113. C
114. C      NCONTE = 0
115. C      NCONTA = 0
116. C
117. C      IF (ANY(EN=Q) GO TO 55
118. C      ASTATS(1) = 0
119. C      ASTATS(2) = 0
120. C
121. C      IF (NACPAR = 54) Q) GO TO 40
122. C
123. C      DO 35 I=1, NACPAR
124. C      ICOD = IABS(IACPARS(I,1)) - 1)
125. C      ISUB = 1
126. C      IF (ICOD .LE. 35) GO TO 30
127. C      ISUB = 2
128. C      ICOD = IABS(ICOD - 34)
129. C
130. C      IF (ICOD .LT. 34) GO TO 30
131. C      WRITE (6,28) ACPARS(1,1)
132. C      28 FORMAT (' ILLEGAL ACOUSTIC CODE', 110)
133. C      GO TO 24
134. C      30 FLD(ICOD,1,ASTATS(ISUB)) = 1
135. C      35 CONTINUE
136. C
137. C      40 ASTATS(1) = 0
138. C      ASTATS(2) = 0
139. C
140. C      IF (NENPAR .EQ. 0) GO TO 55
141. C
142. C      DO 50 I=1, NENPAR
143. C      ICOD = IABS(ENPARS(I,1)) - 1011
144. C      ISUB = 1
145. C      IF (ICOD .LE. 35) GO TO 45
146. C      ISUB = 2
147. C      ICOD = IABS(ICOD - 34)
148. C
149. C      IF (ICOD .LT. 34) GO TO 45
150. C      WRITE (6,43) ENPARS(I,1)
151. C      43 FORMAT (' ILLEGAL ENVIRONMENTAL CODE', 110)
152. C      GO TO 24
153. C      45 FLD(ICOD,1,ESTATS(ISUB)) = 1
154. C      50 CONTINUE
155. C
156. C      55 CALL NTRAN(11,10)
157. C
158. C      CALL NTRAN(11,422,MLAT,L)
159. C      CALL NTRAN(11,22)
160. C
161. C      IF (L = 5) Q) GO TO 45
162. C
163. C      WRITE ERROR
164. C
165. C      WRITE (6,60) L
166. C      60 FORMAT (' ERROR -- WRITING /METREV/ TO TEMP FILE', 110)
167. C      STOP
168. C

```

```

169.      65 FLD(18,18,SECDR) = FLD( 0,18,PNATC)
170.      FLD(18,18,NSEC) = FLD(18,18,PNATC)
171.      C
172.      CALL DSXRED (SECDR,NSEC,DATA)
173.      C
174.      CALL UNPKXP (DATA)
175.      C
176.      C      CHECK AREA, DATE AND STATUS BITS FOR EXPERIMENT LEVEL RECORD
177.      C
178.      IF (CRUO.NE.EXP.AND.EXP.NE.-99999999) GO TO 70
179.      IF (ANY.EQ.0) GO TO 80
180.      C
181.      C      CHECK FOR ENVIRONMENTAL ONLY
182.      C
183.      IF (DATYPE.EQ.1) GO TO 67
184.      C
185.      CALL AREACK (INLAT,ELOW,SLAT,WLON,EAFLAT,EALON,EAFLON,S70)
186.      CALL DATECK (EADAT,EAFLAT,0,0,S70)
187.      CALL STATCK (ASTATS,EAFLAT,S70)
188.      GO TO 49
189.      C
190.      67 CALL AREACK (INLAT,ELOW,SLAT,WLON,EEFLAT,EEFLON,S70)
191.      CALL DATECK (EADAT,EEFLAT,0,0,S70)
192.      C
193.      69 CALL STATCK (ESTATS,EEFLAT,S70)
194.      C
195.      C      CONTINUE ON
196.      C
197.      C      GO TO 80
198.      C
199.      C
200.      70 IF (PNATC.EQ.0) GO TO 200      0 FINISHED PASS THRU DATA BANK
201.      C
202.      GO TO 65
203.      C
204.      C
205.      C
206.      80 PNATC = PCMU
207.      C
208.      85 FLD(18,18,SECDR) = FLD( 0,18,PNATC)
209.      FLD(18,18,NSEC) = FLD(18,18,PNATC)
210.      C
211.      CALL DSXRED (SECDR,NSEC,DATA)
212.      C
213.      C      UNPACK IT
214.      C
215.      CALL UNPKXP (DATA)
216.      C
217.      C
218.      IF (CRUO.NE.CRU.AND.CMU.NE.-99999999) GO TO 90
219.      IF (ANY.EQ.0) GO TO 100
220.      C
221.      C      CHECK FOR ENVIRONMENTAL ONLY
222.      C
223.      IF (DATYPE.EQ.1) GO TO 67
224.      CALL AREACK (INLAT,ELOW,SLAT,WLON,CAFLAT,CAFLON,S70)
225.      CALL DATECK (EADAT,CAFLAT,0,0,S70)
226.      CALL STATCK (ASTATS,CAFLAT,S70)

```

```

226.      C      GO TO 88
227.
228.      C      87 CALL AREACKINLAT,ELON,SLAT,WLON,CELAT,CELON,CEFLAT,CEFLON,990)
229.      CALL DATECKICEIDAT,CEPDAT,0.0,990)
230.
231.      C
232.      C      88 CALL STATCKIESTATS,CESTAT,990)
233.      C
234.      C      CONTINUE ON
235.      C
236.      C      GO TO 100
237.
238.      C
239.      C      90 IF IPNATC .EQ. 01 GO TO 70  0 FINISHED WITH CRUISE LEVEL
240.      C      GO TO 85
241.      C
242.      C
243.      C
244.      C      100 PNATAS = PACS
245.      C
246.      C      CHECK FOR ENVIRONMENTAL ONLY
247.      C
248.      C      IF IDATYPE.EQ.01 GO TO 105
249.      PNATAS=PNENS
250.      ASTATS(1)=ESTATS(1)
251.      ASTATS(2)=ESTATS(2)
252.      NACDCR=NEWDCK
253.      NACPAR=HENPAK
254.      C
255.      C      DO 101 I=1,10
256.      ACDCRS(1) = ENDCRS(1)
257.      NAMIND(1) = NEMIND(1)
258.      DO 101 J = 1,10
259.      AMINDS(1,J) = EMINDS(1,J)
260.      101 CONTINUE
261.      C
262.      C      DO 102 I=1,4
263.      DO 102 J=1,14
264.      102 ACAPRS(1,J)=ENPARS(1,J)
265.      C
266.      C      105 FLD(18,18,SECDRI) = FLD(18,18,PNATAS)
267.      FLD(18,18,NSEC1) = FLD(18,18,PNATAS)
268.      TEMP = PNATAS
269.      C
270.      C      CALL DSKRED (SECADR,NSEC,IBFR)
271.      C
272.      C      UNPACK IT
273.      C
274.      C      CALL ASUNPK (IBFR,TYPE)
275.      C
276.      C      IF ISAND.NE.STA.AND.STA.NE.--99999999) GO TO 110
277.      IF IANT.EQ.0) GO TO 120
278.      CALL AREACKINLAT,ELON,SLAT,WLON,CELAT,CELON,SAFLAT,SAFLON,5110)
279.      CALL DATECKISADAT,SAPDAT,0.0,5110)
280.      CALL STATCKIASATS,SASTAT,5110)
281.      C
282.      C

```

```

283. C CONTINUE ON
284. C
285. C GO TO 120
286. C
287. C 110 IF (PNXTAS.NE.C) GO TO 105 @ HEAD NEXT ACOUSTIC STATION
288. C
289. C GO TO 90
290. C
291. C 120 PNEXTS = PNXTAS
292. C
293. C
294. C EFLAG = 0
295. C
296. C CHECK IF ACOUSTIC OR ENVIRONMENTAL ONLY
297. C IF (RUNTYP.NE.J) GO TO 155
298. C
299. C
300. C
301. C FLD(18,18,SECDR) = FLD(0,18,PENVS)
302. C FLD(18,18,NSEC) = FLD(18,18,PENVS)
303. C TEMP = PENVS
304. C
305. C CALL DSKRED (SECDR,NSEC,18FN)
306. C
307. C UNPACK IT
308. C
309. C CALL ASUHPK (18FN,TYPE)
310. C
311. C IF (ANY.EQ.J) GO TO 135
312. C CALL STATC(ESTATS,SASTAT,S100)
313. C GO TO 135
314. C
315. C 130 PNXTAS = PNEXTS
316. C GO TO 110
317. C
318. C REMIND 12 & INIT @ ENV RUNS COUNTER
319. C
320. C 135 CALL MTRN (12,1C)
321. C EFLAG = 0
322. C IPTR = 11
323. C
324. C INITIALIZE TEMPORARY ENV. COUNTERS
325. C
326. C DO 133 I = 1,NENDT
327. C 133 TMPD(I) = 0
328. C DO 134 I = 1,NEMPT
329. C TMPDST(I) = 0
330. C 134 TMPPT(I) = 0
331. C
332. C
333. C LOOP TO EXAMINE RUNS
334. C DO 150 I=1,SANRHN
335. C
336. C OPTR = IPTR
337. C
338. C CALL AUMPK (18FN,IPTR)
339. C

```



```

340. IF (MUNNG.NL.NUN.AND.NUN.HE..9Y999999) GO TO 150
341. CALL RANGE (IMPAN,IMPAN$,$150,MEMIND,EMIND$,$EQUIV)
342. IF (ANY.EQ.O) GO TO 140
343. CALL STATC (KSTAT$,$STAT,$150)
344. C
345. C CHECK EAV: MODE$
346. C
347. C
348. IF (MEMOCR.EQ.O) GO TO 140
349. DO 136 J=1,MEMOCR
350. IF (MEMOCR(J).EQ. DESCR) GO TO 140
351. 136 CONTINUE
352. GO TO 150
353. C
354. DO 131 J=1,MOSEA
355. IF (SEX(J).EQ.O) GO TO 142
356. IF (ILLAS.EQ.SERJ11) GO TO 142
357. 141 CONTINUE
358. GO TO 150
359. C
360. C WRITE DATA TO 12
361. 142 TOT11111 = SAND
362. TOT11112 = TEMP
363. TOT11113 = QPTR
364. TOT11114 = MOUSED
365. DO 143 J=1,50
366. TOT1111(J+4) = NRS(J)
367. 143 CONTINUE
368. C
369. CALL NTRAN (12,1,54,TOT1111)
370. CALL NTRAN (12,22)
371. IF (L.EQ. 54) GO TO 147
372. C
373. C ERROR
374. C
375. WHITE (A,144) L
376. 144 FORMAT (1,ERROR= WRITE TO 12,110)
377. STOP
378. C
379. C INCREMENT COUNTER
380. C
381. C
382. 147 EFLAG = EFLAG + 1
383. C
384. TMPDT(DESCN) = TMPDT(DESCN) + 1
385. C
386. IF (INCON.EQ.O) GO TO 1470
387. DO 1480 J=1,INCON
388. INDEX = ICON(1,J) - 100
389. 1480 TMPPT(INDEX) = TMPPT(INDEX) + 1
390. 1470 IF (INPAR.EQ.O) GO TO 1470
391. DO 1480 J=1,INPAR
392. INDEX = IPAN(2,J) - 100
393. 1480 TMPPT(INDEX) = TMPPT(INDEX) + 1
394. 1480 TMPDT(INDEX) = TMPDT(INDEX) + MOUSED
395. 1490 IF (INVAR.EQ.O) GO TO 150
396. DO 1495 J=1,INVAR

```

```

397. INDEX = IVAN(I,J) - 100
398. TMPPT(I,INDEX) = TMPPT(I,INDEX) + 1
399. 1499 TMPDST(I,INDEX) = TMPDST(I,INDEX) + HOUSED
400. C 150 CONTINUE
401. C
402. C CHECK FOR NONE FOUND
403. C
404. C
405. C IF (EFLAG.EQ.0) GO TO 150 W GO GET NEXT ACOUSTIC STATION
406. C
407. C READ BACK IN ACOUSTIC RUN
408. C FLD(18,18,SECADR) = FLD(0,18,PENVS)
409. C FLD(18,18,MSEC) = FLD(18,18,PENVS)
410. C TEMP = PENVS
411. C
412. C CALL DSKRD (SECADR,MSEC,18FR)
413. C
414. C UNPACK IT
415. C
416. C CALL ASUNPK (18FR,TYPE)
417. C
418. C SET COUNTER AND BUFFER POINTER
419. C
420. C
421. C 155 AFLAG = 0
422. C IPTR = 11
423. C INIT FOR THIS STATION OUTPUT TO II
424. C
425. C 101111) = EANO
426. C 101112) = CRUM0
427. C 101113) = SANO
428. C 101114) = TEMP
429. C
430. C LOOP FOR PROCESSING ACOUSTIC RUNS
431. C DO 170 I=1,SANRUM
432. C
433. C SET OLD PTR = NEW
434. C
435. C
436. C OPTR = IPTR
437. C
438. C UNPACK RUN
439. C
440. C CALL AUNPK (18FR,IPTR)
441. C
442. C IF (KUNNO.NE.RUN.AND.RUN.NE.-99999999) GO TO 170
443. C CALL RANGE (INACPAR,ACPARS,8170,MANTHO,AMTHDS,AEQUIV)
444. C IF (ANY.EQ.0) GO TO 161
445. C TILAT = MAX (TILAT,REFLAT)
446. C TFLAT = MIN (TILAT,REFLAT)
447. C TILON = EAST (TILON,REFLON)
448. C TFLON = WEST (TILON,REFLON)
449. C CALL AREACK (INLAT,TELON,SLAT,HLON,TILAT,TILON,TFLAT,TFLON,8170)
450. C CALL DATECK (IRIDAY,RFDAY,RTIM,RTIM,8170)
451. C CALL STATCK (ASTATS,RSTAT,817C)
452. C
453. C PRINT 159,DESCR,1ACORST(J),J=1,NACDER)

```

```

454. C 159 FORMATION
455. IF (INACDCH.EQ.O) GO TO 156
456. DO 160 J=1,NACDCH
457. IF (INACDCH(I).EQ.DESCH) GO TO 156
458. 160 CONTINUE
459. GO TO 170
460. C
461. C
462. C
463. C
464. C
465. C
466. C
467. C
468. C
469. C
470. C
471. C
472. C
473. C
474. C
475. C
476. C
477. C
478. C
479. C
480. C
481. C
482. C
483. C
484. C
485. C
486. C
487. C
488. C
489. C
490. C
491. C
492. C
493. C
494. C
495. C
496. C
497. C
498. C
499. C
500. C
501. C
502. C
503. C
504. C
505. C
506. C
507. C
508. C
509. C
510. C

      SKIP SOURCE/RECEIVER CHECKS IF ENVIRONMENTAL

154 IF (DATYPE.EQ.1) GO TO 161
      CHECK SOURCE TYPE
      C
      IF (INSTR.EQ.O) GO TO 150
      DO 157 J=1,NINSTR
      IF (INSTR(I).EQ.SRC) GO TO 150
      157 CONTINUE
      GO TO 170
      C
      CHECK RECEIVER TYPE
      C
      IF (INSTR.EQ.O) GO TO 164
      DO 163 J=1,NINSTR
      IF (INSTR(I).EQ.RC) GO TO 164
      163 CONTINUE
      GO TO 170
      C
      CHECK SOURCE SYSTEM
      C
      IF (INSTR.EQ.O) GO TO 164
      DO 166 J=1,NINSTR
      IF (INSTR(I).EQ.SSRC) GO TO 166
      166 CONTINUE
      GO TO 170
      C
      CHECK RECEIVER SYSTEM
      C
      IF (INSTR.EQ.O) GO TO 161
      DO 171 J=1,NINSTR
      IF (INSTR(I).EQ.SSRC) GO TO 161
      171 CONTINUE
      GO TO 170
      C
      CHECK FOR SECURITY
      C
      DO 167 J=1,NUSER
      IF (SEC(I).EQ.O) GO TO 169
      IF (CLASS.EQ.SEL(I)) GO TO 169
      167 CONTINUE
      GO TO 170
      C
      169 TOT(15) = OPTA
      TOT(16) = HOUSE
      DO 1610 J=1,50
      TOT(17+J) = NMS(J)
      1610 CONTINUE

```

```

511. C      CALL NTRAN (I1,I1,S6,I0I1,I1)
512.      CALL NTRAN (I1,I2)
513.      IF IL .EQ. 54) GO TO 145
514.      WRITE (4,142) L1,I,SAMRUN
515.      142 FORMAT (1X,ERROR IN ACOUSTIC WRITE TO I1',J1I1Q)
516.      STOP
517.
518. C      INCREMENT ACOUSTIC ON ENV. ONLY COUNTER
519. C
520. C      145 AFLAG = AFLAG + 1
521. C
522. C      INCREMENT DESCRIPTION AND PARAMETER TABLE COUNTERS
523. C
524. C      BRANCH IF ENV. ONLY
525. C
526. C
527. C      IF (DATYPE.EQ.1) GO TO 1750
528. C
529. C      ACOTIDESCR) = ACOTIDESCR) + 1
530. C
531. C      IF (INCOM.EQ.0) GO TO 1670
532. C      DO 1666 J = 1,INCOM
533. C      INDEX = ICON(I,J)
534. C      1666 ACPT(INDEX) = ACPT(INDEX) + 1
535. C      1670 IF (INPAR.EQ.0) GO TO 1690
536. C      DO 1680 J = 1,INPAR
537. C      INDEX = IPAR(I,J)
538. C      ACPT(INDEX) = ACPT(INDEX) + 1
539. C      1680 ACPOST(INDEX) = ACPOST(INDEX) + NDUSED
540. C      1690 IF (INVAR.EQ.0) GO TO 170
541. C      DO 1695 J = 1,INVAR
542. C      INDEX = IVAR(I,J)
543. C      ACPT(INDEX) = ACPT(INDEX) + 1
544. C      1695 ACPOST(INDEX) = ACPOST(INDEX) + NDUSED
545. C      GO TO 170
546. C
547. C      INCREMENT ENV. ONLY COUNTERS
548. C
549. C      1750 ENDTIDESCR) = ENDTIDESCR) + 1
550. C
551. C      IF (INCOM.EQ.0) GO TO 1770
552. C      DO 1760 J = 1,INCOM
553. C      INDEX = ICON(I,J) - 100
554. C      1760 ENPT(INDEX) = ENPT(INDEX) + 1
555. C      1770 IF (INPAR.EQ.0) GO TO 1790
556. C      DO 1780 J = 1,INPAR
557. C      INDEX = IPAR(I,J) - 100
558. C      ENPT(INDEX) = ENPT(INDEX) + 1
559. C      1780 EMPST(INDEX) = EMPST(INDEX) + NDUSED
560. C      1790 IF (INVAR.EQ.0) GO TO 170
561. C      DO 1795 J = 1,INVAR
562. C      INDEX = IVAR(I,J) - 100
563. C      ENPT(INDEX) = ENPT(INDEX) + 1
564. C      1795 EMPDST(INDEX) = EMPDST(INDEX) + NDUSED
565. C
566. C      170 CONTINUE
567. C

```



```

568. C
569. C CHECK FOR NO ACOUSTIC MUNS FOUND
570. C
571. C IF (AFLAG.EQ. 0) GO TO 110
572. C
573. C INCREMENT ENV: COUNTS FROM TEMPORARY COUNTS
574. C
575. C IF (RUNTIME.NE.0) GO TO 1498
576. C DO 1496 J = 1,NENDT
577. C 1496 ENDT(J) = ENDT(J) + TMPDT(J)
578. C GO 1497 J = 1,NENDT
579. C ENDT(J) = ENDT(J) + TMPDT(J)
580. C 1497 ENPOST(J) = ENPOST(J) + TMPDST(J)
581. C
582. C REWIND 12
583. C
584. C 1498 CALL NTRAN (12,10)
585. C
586. C CHECK EFLAG
587. C
588. C IF (EFLAG.EQ. 0) GO TO 190
589. C
590. C LOOP TO MOVE ENV RUN POINTERS TO 11
591. C
592. C DO 190 I=1,EFLAG
593. C
594. C READ FROM 12
595. C
596. C CALL NTRAN (12,2,54,10111(3),1)
597. C CALL NTRAN (12,22)
598. C IF (L.EQ. 54) GO TO 174
599. C STOP ERR 10
600. C
601. C WRITE TO 11
602. C
603. C 174 CALL NTRAN (11,1,56,10111,1)
604. C CALL NTRAN (11,22)
605. C IF (L.EQ. 56) GO TO 190
606. C WRITE (6,175) L,EFLAG
607. C 175 FORMAT (' ERROR IN ENV WRITE TO 11',31101)
608. C STOP
609. C
610. C 180 CONTINUE
611. C
612. C
613. C 190 NCONTA = NCONTA + AFLAG
614. C NCONTE = NCONTE + EFLAG
615. C GO TO 110
616. C
617. C
618. C CHECK FOR ENVIRONMENTAL ONLY
619. C
620. C 200 IF (DATEP.EQ.0) GO TO 203
621. C TEMP=NCONTA
622. C NCONTA=NCONTE
623. C NCONTE=TEMP
624. C

```

BEST AVAILABLE COPY

```

625. C PRINT RESULTS
626. C
627. 203 WRITE (6,201) NCONT,NCONTE
628. 201 FORMAT (//,'THESE ARE:',15,' ACUSTIC RUN(S)',19,'AND',15,' ENV'
629. 'IRONMENTAL RUN(S) WHICH MEET SELECTION CRITERIA')
630. C
631. C UPDATE COMMON RETNEV ON DISK...
632. C
633. CALL MTRAN (11,10)
634. NPKTS = NCONTE + NCONTE
635. CALL MTRAN (11,1,422,NLAT,L)
636. CALL MTRAN (11,22)
637. IF (L.EQ. 422) GO TO 204
638. WRITE (6,331) L
639. 331 FORMAT ('ERROR WRITING NPKTS TO 11',110)
640. STOP
641. C
642. C DISPLAY MENU OPTIONS
643. C
644. C IF SUMMARY DATA LISTING APPLIC. NUMBER IS CHANGED, CALL SHORT STATEMENT IN
645. C DATLIST MUST ALSO BE CHANGED.
646. 204 IF (LISTF) GO TO 208
647. LISTF = .TRUE.
648. 213 WRITE (6,205)
649. 205 FORMAT (//,'10. NAVDAG APPLICATION OPTIONS',/,/,
650. ' 1 - SPECIFY SELECTION CRITERIA',/,
651. ' 2 - INVENTORY OF RUNS SATISFYING SELECTION CRITERIA',/
652. ' 3 - PRINT THE NOTES',/
653. ' 4 - SUMMARY DATA LISTING BY RUN',/
654. ' 5 - COMPLETE DATA LISTING BY RUN',/
655. ' 6 - ACOUSTIC DESCRIPTOR INVENTORY SATISFYING SELECTION',/
656. ' 7 - ENVIRONMENTAL DESCRIPTOR INVENTORY SATISFYING SELE',/
657. ' CTION',/
658. ' 8 - ACOUSTIC PARAMETER INVENTORY SATISFYING SELECTION',/
659. ' 9 - ENVIRONMENTAL PARAMETER INVENTORY SATISFYING SELEC',/
660. ' TION',/
661. ' 10 - SINGLE UNIT CONVERSIONS',/
662. ' 11 - EXECUTE USER APPLICATION PROGRAM',/)
663. C
664. C READ INPUT
665. C
666. C
667. C
668. 208 WRITE (6,209)
669. 209 FORMAT (//,' ENTER APPLICATION OPTION NUMBER / ENTER 'LISTY' / ENY
670. 'ER 'DONE',)
671. 202 READ (5,211,END=208,ERR=210) (CHAR(1),1=1,80)
672. 211 FORMAT (80A1)
673. WRITE (6,207) (CHAR(1),1=1,80)
674. FORMAT (3X,80A1)
675. CALL FREEFD (CHAR,WORD,APPL,NWORD,5,1,3210)
676. IF (INWORDS.EQ.0) GO TO 210
677. W = ' '
678. FLD(3,6,0) = FLD(0,6,APPL)
679. IF (W.EQ.'L') GO TO 213
680. IF (A.EQ.'D') GO TO 340
681. GO TO 250

```

```

682. C
683. C
684. C
685. C
686. C
687. C
688. C
689. C
690. C
691. C
692. C
693. C
694. C
695. C
696. C
697. C
698. C
699. C
700. C
701. C
702. C
703. C
704. C
705. C
706. C
707. C
708. C
709. C
710. C
711. C
712. C
713. C
714. C
715. C
716. C
717. C
718. C
719. C
720. C
721. C
722. C
723. C
724. C
725. C
726. C
727. C
728. C
729. C
730. C
731. C
732. C
733. C
734. C
735. C
736. C
737. C
738. C

      ERRON ON READ

215. WRITE (6,215)
216. FORMAT (1, '%% READ FORMAT ERRON .....')
217. GO TO 208
220. WRITE (6,221) APPL
221. FORMAT (1, '%% ENTERED NUMBER', 15, ' OUT OF RANGE .....')
222. GO TO 208

250. IF (APPL.E-Q-UM-APPL-61.11) GO TO 220
251. GO TO (110,280,290,320,330,350,360,370,380,390,330),APPL

      INVENTORY PROGRAM

280. CALL INVTY
281. GO TO 208

      NOTES PROGRAM

290. CALL NOTE (DATA)
291. GO TO 208

      LIST PROGRAM

300. CALL DATLST (APPL,CHAR,NUMTYP)
301. GO TO 208

      UNIT CONVERSION

320. CALL CVTPRO
321. GO TO 208

      EXECUTE USER PGM

330. WRITE (6,331)
331. FORMAT (120, 'ENTER EXQT USERS PROGRAM')
332. STOP

340. WRITE (6,341)
341. FORMAT (114, 'MAYDAY RETRIEVAL SYSTEM -- SESSION COMPLETE')
342. STOP

      ACOUSTIC DESCRIPTION TABLE

350. WRITE (6,351)
351. FORMAT (11, 'ACOUSTIC DESCRIPTION INVENTORY SATISFYING SELECT')
352. STOP

      CALL DESCIB (ACUT,MACOT,8)
353. GO TO 208

      ENVIRONMENTAL DESCRIPTION TABLE

```

```

739.      WRITE (6,365)
740.      JCS  FORMAT (11,'ENVIRONMENTAL DESCRIPTION INVENTORY SATISFYING SEL'
741.      1'ACTION',1)
742.      CALL DESCIB (ENDT,MENDT,7)
743.      GO TO 208
744.
745.      C      ACOUSTIC PARAMETER TABLE
746.      C
747.      WRITE (6,375)
748.      JCS  FORMAT (11,'ACOUSTIC PARAMETER INVENTORY SATISFYING SELECTION'
749.      1'N',1)
750.      CALL PARATB (ACPT,ACPDST,MACPT,2)
751.      GO TO 209
752.
753.      C      ENVIRONMENTAL PARAMETER TABLE
754.      C
755.      WRITE (6,385)
756.      JCS  FORMAT (11,'ENVIRONMENTAL PARAMETER INVENTORY SATISFYING SELE'
757.      1'CTION',1)
758.      CALL PARATB (EMPT,EMPDST,MENDP,5)
759.      GO TO 208
760.
761.      C      END

```


9F04,5# H.SHORT, .SHORT

```

1: SUBROUTINE SHORT (NMS,DATYPE,IVLRNG,UNITS,NAME,S)
2: C
3: C SHUNT PROVIDES A SUMMARY DATA LISTING OF THE RUNS MEETING THE USER'S
4: C REQUIREMENTS. THIS IS IDENTICAL TO THE LONG LISTING EXCEPT THE RANGE OF
5: C THE VARIABLES ARE PRINTED INSTEAD OF THE ACTUAL VALUES. LONG RUN LISTING
6: C STATEMENTS ARE BYPASSED BY USING RETURN 6.
7: C
8: C IMPLICIT INTEGER (A-Z)
9: C
10: C REAL VALRNG,VALMIN,VALMAX
11: C DIMENSION IVLRNG(2,30,50),UNITS(1),NAME(1)
12: C COMMON /CPV /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
13: C NDATS,MNOMS(50),NDATA,VALRNG(2,30,50),DATA(3,500)
14: C
15: C PRINT HEADING FOR VARIABLES
16: C
17: C IF (DATYPE.NE.3) GO TO 10
18: C WRITE (6,5)
19: C FORMAT (7/25,'MINIMUM MAXIMUM',160,'ACOUSTIC')
20: C GO TO 20
21: C WRITE (6,15)
22: C FORMAT (7/25,'MINIMUM MAXIMUM',150,'ENVIRONMENTAL')
23: C WRITE (6,30)
24: C FORMAT (7/25,'NAME',126,'VALUES',137,'VALUE',140,'UNITS',161,
25: C 1'METHOD')
26: C
27: C LOOP THROUGH VARIABLES FOR DATA SET NRS
28: C
29: C DO 50 I = 1,NVAR
30: C CALL NAMEGT (IVAR(I,1),NAME)
31: C VALMIN = VALRNG(1,1,NRS)
32: C VALMAX = VALRNG(2,1,NRS)
33: C CALL CHECK (VALMIN,IVAR(I,1),UNITS)
34: C CALL CHECK (VALMAX,IVAR(I,1),UNITS)
35: C WRITE (6,40) (NAME(J),J=1,2),VALMIN,VALMAX,UNITS(1),UNITS(2),
36: C 1,IVAR(I,1)
37: C FORMAT (11,2A6,6I2,5,135,6I1,5,147,2A6,2X,14)
38: C CONTINUE
39: C WRITE (6,63)
40: C FORMAT (777)
41: C RETURN 6
42: C
43: C
44: C
45: C
46: C
47: C
48: C
49: C
50: C
51: C

```

BF03,56

N.SHRTIME,,SHRTIME

```

1.      SUBROUTINE SHRTIME(VALUE,UNICODE,UNITS)
2.      C
3.      C      TABLE OF SHRTIME MEASURE CONVERSION FACTORS.
4.      C
5.      C      INTEGER ENTRY(3,5),UNITS(4),UNICODE
6.      C
7.      C      FACTORS ARE DOUBLE PRECISION.
8.      C
9.      C      DOUBLE PRECISION FACTOR(5)
10.     C
11.     C      PUT UNIT CODES INTO ENTRY.
12.     C
13.     C      DATA (ENTRY(1,1),1=1,5)/21,22,23,24,25/
14.     C
15.     C      PUT IN ALPHA UNITS.
16.     C
17.     C      DATA (ENTRY(1,J),J=2,3),J=1,5)/'MILLIS','SECONDS',
18.     C      'SECOND','S', 'MINUTE','M',
19.     C      'HOURS','H', 'DAYS','D', '/'
20.     C
21.     C      ENTER INTERNAL UNITS.
22.     C
23.     C      UNITS(3)='SEC', '
24.     C      UNITS(4)=' '
25.     C
26.     C      IF UNICOD=0, THEN SET CODE TO STANDARD UNITS.
27.     C
28.     C      IF(UNICOD.EQ.0) UNICOD=-22
29.     C
30.     C      SET CONVERSION FACTORS.
31.     C
32.     C      DATA (FACTOR(1),1=1,5)/.00100,1.000,60.00,3.600,8.6400/
33.     C
34.     C      IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
35.     C      CODE INTO 'UNITS'.
36.     C
37.     C      ICODE=IABS(UNICODE)
38.     C      DO 2 I=1,5
39.     C      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
40.     C      IF(UNICODE.GE.0) VALUE=VALUE*FACTOR(I)
41.     C      IF(UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
42.     C      DO 1 J=1,2
43.     C      UNITS(J)=ENTRY(J,1)
44.     C      1 CONTINUE
45.     C
46.     C      AT THIS POINT, CONVERSION IS COMPLETE.
47.     C
48.     C      RETURN
49.     C      2 CONTINUE
50.     C
51.     C      IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
52.     C
53.     C      UNITS(1)='UNITS '
54.     C      UNITS(2)='ERROR '

```

55. PRINT I31,ICODE
56. I31 FORMATTED ERROR--(UNIT CODE',I31,' NOT IN SHORT TIME TABLE.')

57. RETURN
58. END

W. SORGET, SORGET

```

SUBROUTINE SOURCE(VALU,UNICODE,UNITS)
  TABLE OF SOURCE LEVEL/TARGET STRENGTH CONVERSION FACTORS.

  INTEGER ENTRY(3,3),UNITS(4),UNICODE
  FACTORS ARE DOUBLE PRECISION.
  DOUBLE PRECISION FACTOR(3)
  PUT UNIT CODES INTO ENTRY.

  DATA (ENTRY(1,1),1,1,1,3)/4,65,91/
  PUT IN ALPHA UNITS.

  DATA ((ENTRY(1,1),1,1,1,3)/2,1,3)/08//1U//BARGIN//
  '08//1U//BARGIN//08//1U//PASQIN//
  *
  ENTER INTERNAL UNITS.
  UNITS(3)='08//1U'
  UNITS(4)='PASQIN'

  CODE TO PACK DB//IERG/CM2HI YD.

  IF (UNICODE .NE. 96) GO TO 5
  FLD(18,18,VALUE) = VALUE
  FLD( 0,18,VALUE) = -UNICODE
  UNITS(1) = '//ERG/'
  UNITS(2) = 'CM2HYD'
  RETURN

  CODE TO UNPACK DB//IERG/CM2HI YD.

  5 IF (UNICODE .NE. -96) GO TO 6
  UNICDE = FLD( 3,18,VALUE)
  VALUE = FLD(18,18,VALUE)
  UNITS(1) = '//ERG/'
  UNITS(2) = 'CM2HYD'
  RETURN

  IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.

  6 IF(UNICDE.EQ.0) UNICDE=-91

  SET CONVERSION FACTORS.

  DATA (FACTOR(1),1,1,3)/-120,300,-99,200,000/
  INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND
  UNITS(1)='UNITS '

```



```

55.      UNITS(2)=ERROR
56.      C
57.      IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
58.      C
59.      C
60.      C
61.      C
62.      C
63.      C
64.      C
65.      C
66.      C
67.      C
68.      C
69.      C
70.      C
71.      C
72.      C
73.      C
74.      C
75.      C
76.      C
77.      C
78.      C
79.      C
80.      C
81.      C

```

UNITS(2)=ERROR
 IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
 CODE INTO 'UNITS'.
 ICODE=TABLE(UNICODE)
 DO 2 1=1,3
 IF (ICODE.NE.ENTRY(1,1)) GO TO 2
 IF (UNICODE.GE.0) VALUE=VALUE*FACTOR(1)
 IF (UNICODE.LT.0) VALUE=VALUE*-FACTOR(1)
 UNITS(1)=ENTRY(2,1)
 UNITS(2)=ENTRY(3,1)
 AT THIS POINT, CONVERSION IS COMPLETE.
 RETURN
 ELSE, CHECK REST OF TABLE.
 2 CONTINUE
 IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
 PRINT ICI, ICODE
 131 FORMAT(' ERROR--UNIT CODE',13,' NOT IN SOURCE/TARGET TABLE.')

OFURIS? N,SPEEDS,,SPEEDS

```

1. SUBROUTINE SPEEDS(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF SPEEDS MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(I,J)=1.0/UNIT(I,J),UNICODE
6. C
7. C FACTORS ARE DOUBLE PRECISION.
8. C
9. C DOUBLE PRECISION FACTOR(I)
10. C
11. C PUT UNIT CODES INTO ENTRY.
12. C
13. C DATA (ENTRY(I,J),I=1,8)/20,29,30,31,32,33,34,99/
14. C
15. C PUT IN ALPHA UNITS.
16. C
17. C DATA (ENTRY(I,J),I=2,3)/1,8)/'CM/SEC',,
18. C 'METERS',/SEC, 'FT./SEC.',,
19. C 'YARDS',/SEC, 'KNOTS',,
20. C 'KM./HR.',, 'MILES',/HR.,,
21. C 'FURLONGS',/FURLONG,
22. C
23. C ENTER INTERNAL UNITS.
24. C
25. C UNITS(3)='METERS'
26. C UNITS(4)='Y./SEC.'
27. C
28. C IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
29. C
30. C IF (UNICODE.EQ.0) UNICODE=-29
31. C
32. C SET CONVERSION FACTORS:
33. C
34. C DATA (FACTOR(I),I=1,8)/.0100,100,.304800,.914400,.514500,.27777703
35. C ,.446944400,1.6620-4/
36. C
37. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
38. C CODE INTO 'UNITS'.
39. C
40. C ICODE=ABS(UNICODE)
41. C DO 2 I=1,8
42. C IF (ICODE.EQ.ENTRY(I,I)) GO TO 2
43. C IF (UNICODE.EQ.0) VALUE=VALUE*FACTOR(I)
44. C IF (UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
45. C DO 1 J=1,2
46. C UNITS(J)=ENTRY(J,I,1)
47. C I CONTINUE
48. C
49. C AT THIS POINT, CONVERSION IS COMPLETE.
50. C
51. C RETURN
52. C
53. C ELSE, CHECK NEXT OF TABLE.
54. C

```

```

55.      2 CONTINUE
56.      C
57.      C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
58.      C
59.      UNITS(1)='UNITS '
60.      UNITS(2)='ERROR '
61.      PRINT 101,ICODE
62.      I=1 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN VELOCITY TABLE.')
63.      RETURN
64.      END

```

QELT,L N,STATCK,STATCK

```
1. SUBROUTINE STATCK (MASK,STATUS,*)
2. C
3. IMPLICIT INTEGER (A-Z)
4. DIMENSION MASK(2),STATUS(2)
5. C
6. C
7. DO IC=1,2
8. CK = AND(MASK(1),STATUS(1))
9. IF (CK .NE. MASK(1)) RETURN 3
10. CONTINUE
11. C
12. RETURN
13. C
14. END
```


QELT,L N.STORE,.,STONE

```

1: SUBROUTINE STONE
2:
3: C THIS PROGRAM STORES COMMON/RETRIEV/ INTO THE USAGE FILE (UNIT USGUNT).
4:
5: C
6: C
7: C
8: C
9: C
10: C
11: C
12: C
13: C
14: C
15: C
16: C
17: C
18: C
19: C
20: C
21: C
22: C
23: C
24: C
25: C
26: C
27: C
28: C
29: C
30: C
31: C
32: C
33: C
34: C
35: C
36: C
37: C
38: C

```

PARAMETER USGUNT = 14
INCLUDE RETREV,LIST
IMPLICIT INTEGER (A-Z)
HEAD SECTION 1 OF USAGE FILE
CALL NTRAN (USGUNT,10,2,1,LOC,LSTAT,22)
IF (LSTAT.GT.0) GO TO 20
ERROR READING USAGE FILE
WRITE (6,10) LSTAT
FORMAT (1,'... ERROR READING USAGE FILE - ERROR CODE',14,'...')
STOP ERROR
WRITE THE RETRIEVAL REQUESTS TO THE USAGE FILE
CALL SETADR (USGUNT,LOC)
CALL NTRAN (USGUNT,1,428,NLAT,LSTAT,22)
IF (LSTAT.GT.0) GO TO 30
ERROR WRITING TO USAGE FILE
WRITE (6,10) LSTAT
FORMAT (1,'... ERROR WRITING TO USAGE FILE - ERROR CODE',14,'...')
STOP ERROR
WRITE NEW LOCATION SPECIFICATION
LOC = LOC + 14
CALL NTRAN (USGUNT,10,1,1,LOC,LSTAT,22)
IF (LSTAT.LE.0) GO TO 30
RETURN
END

QFOR,50 N,TMPTUR,,TMPTUR

```

1. SUBROUTINE TMPTUR(VALUE,UNICODE,UNITS)
2. C
3. C TABLE OF TMPTUR MEASURE CONVERSION FACTORS.
4. C
5. C INTEGER ENTRY(2,4),UNITS(4),UNICODE
6. C
7. C PUT UNIT CODES INTO ENTRY.
8. C
9. C DATA (ENTRY(1,1),1=1,4)/35,36,37,38/
10. C
11. C PUT IN ALPHA UNITS.
12. C
13. C DATA (ENTRY(1,2),1=2,3)/1,4/'FAREN','HEIT '
14. C
15. C 'DEG C','ELSIUS','KELVIN','
16. C 'RANKIN','ES '
17. C
18. C ENTER INTERNAL DEGREE UNITS.
19. C
20. C UNITS(3)=DEGREE
21. C UNITS(4)=5 CENT
22. C
23. C IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
24. C IF (UNICDE.EQ.0) UNICDE=-36
25. C
26. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
27. C CODE INTO 'UNITS'.
28. C
29. C ICODE=IABS(UNICDE)
30. C DO 1 1=1,4
31. C IF (ICODE.NE.ENTRY(1,1)) GO TO 1
32. C GO TO (2,3,4,5),1
33. C 1 CONTINUE
34. C
35. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
36. C
37. C UNITS(1)='UNITS '
38. C UNITS(2)='ERROR '
39. C PRINT 101,ICODE
40. C RETURN
41. C
42. C 2 IF (UNICDE.GE.0) VALUE=(VALUE-32.15)/9.
43. C IF (UNICDE.LT.0) VALUE=VALUE*9.75+32.
44. C UNITS(1)=ENTRY(2,1)
45. C UNITS(2)=ENTRY(3,1)
46. C RETURN
47. C
48. C 3 UNITS(1)=ENTRY(2,1)
49. C UNITS(2)=ENTRY(3,1)
50. C RETURN
51. C
52. C 4 IF (UNICDE.GE.0) VALUE=VALUE-273.15
53. C IF (UNICDE.LT.0) VALUE=VALUE+273.15
54. C UNITS(1)=ENTRY(2,1)
55. C UNITS(2)=ENTRY(3,1)
56. C RETURN
57. C
58. C 5 IF (UNICDE.GE.0) VALUE=(VALUE-49).6715+9.
59. C IF (UNICDE.LT.0) VALUE=VALUE*1.8+32.
60. C UNITS(1)=ENTRY(2,1)
61. C UNITS(2)=ENTRY(3,1)
62. C RETURN
63. C
64. C
65. C
66. C
67. C
68. C
69. C
70. C
71. C
72. C
73. C
74. C
75. C
76. C
77. C
78. C
79. C
80. C
81. C
82. C
83. C
84. C
85. C
86. C
87. C
88. C
89. C
90. C
91. C
92. C
93. C
94. C
95. C
96. C
97. C
98. C
99. C
100. C

```

```

55. IF (UNITCODE.LT.0) VALUE=VALUE*9./5.+491.67
56. UNITS(1)=UNIT(2,1)
57. UNITS(2)=UNIT(13,1)
58. RETURN
59. 101 FORMAT(1, ' ERROR--UNIT CODE ',13,' NOT IN TEMPERATURE TABLE. ')
60. END

```

9ELT,L N,UNPACK,UNPACK

```

1. SUBROUTINE UNPACK(DATA)
2. IMPLICIT INTEGER(A-Z)
3. C
4. DIMENSION DATA(1),INOTES(1)
5. C
6. COMMON /CRUISE/CRUNG,PNTIC,CALAT,CALON,CAIDAT,CAFLAT,CAFLON,
7. CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEPDAT,
8. CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
9. CAFLG
10. C
11. EQUIVALENCE(INOTES(1),NOTES(1,1))
12. C
13. CRUNG=FLD(7,18,DATA(2))
14. C
15. C UNPACK NUMBER OF CARDS.
16. C
17. NCARDS=FLD(10,18,DATA(2))
18. C
19. PNTIC=DATA(3)
20. C
21. C UNPACK ACOUSTIC INITIAL AND FINAL POINT BOUNDS.
22. C
23. C
24. CALL UPKREAL('CALAT,CALON,CAIDAT,0',' ',
25. CAFLAT,CAFLON,CAFDAT,0',' ',DATA(4))
26. C
27. C UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
28. C
29. C
30. CALL UPKREAL('CEILAT,CEILON,CEIDAT,0',' ',
31. CEFLAT,CEFLON,CEFDAT,0',' ',DATA(5))
32. C
33. CASTAT(1)=DATA(12)
34. CASTAT(2)=DATA(13)
35. C
36. CESTAT(1)=DATA(14)
37. CESTAT(2)=DATA(15)
38. C
39. PACS=DATA(16)
40. PENS=DATA(17)
41. C
42. C UNPACK NOTES.
43. C
44. N=14*NCARDS
45. DO 1 I=1,N
46. INOTES(I)=0
47. I INOTES(I)=DATA(1+I7)
48. RETURN
49. END

```


WELT,L N,UNPKAP,UNPKAP

```
1. SUBROUTINE UNPKAP(DATA)
2. IMPLICIT INTEGER(A-Z)
3. C
4. C DIMENSION DATA(1)
5. C
6. COMMON /EXPHN1/EXNO,PNATE,PCRU,EAFLAT,EAFLON,EAIDAT,EAFLAT,EAFLON,
7. * EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
8. * EASTAT(12),EESTAT(12),EAFLO
9. C
10. C
11. C UNPACK EXPERIMENT NUMBLN
12. C
13. C EXNO=DATA(2)
14. C
15. C UNPACK ACQUSTIC INITIAL AND FINAL POINT BOUNDS
16. C
17. C CALL UNPKREAL 'EAFLAT,EAFLON,EAIDAT,0.0,0.0,
18. * EAFLAT,EAFLON,EAFDAT,0.0,0.0,DATA(3))
19. C
20. C UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS
21. C
22. C CALL UNPKREAL 'EEILAT,EEILON,EEIDAT,0.0,0.0,
23. * EEFLAT,EEFLON,EEFDAT,0.0,0.0,DATA(7))
24. C
25. C EASTAT(11)=DATA(11)
26. C EASTAT(12)=DATA(12)
27. C
28. C EESTAT(11)=DATA(13)
29. C EESTAT(12)=DATA(14)
30. C
31. C PNATE=DATA(15)
32. C
33. C PCRU=DATA(16)
34. C RETURN
35. C END
```

REFR,SA H,UNTCVT,UNTCVT

```

1. SUBROUTINE UNTCVT
2.
3. C THIS ROUTINE CONVERTS THE SELECTED CONSTANTS, PARAMETERS, AND
4. C VARIABLES FROM SYSTEM UNITS TO THE REQUESTED UNITS.
5.
6. IMPLICIT INTEGER (A-Z)
7. REAL CON,PAR,DATA
8. DIMENSION ICON(4,30),IPAR(30,30),UNITS(4)
9. INCLUDE CPV.LIST
10. INCLUDE DATCON.LIST
11. EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1))
12.
13. IF (INCON.EQ.C) GO TO 20
14. C
15. C CONVERT THE CONSTANTS
16. C
17. DO 10 I = 1,INCON
18. 10 CALL CHECK (CON(3,I),ICON(1,I),UNITS)
19. C
20. C CHECK IF THERE ARE ANY DATA SETS
21. C
22. 20 IF (INDATS.EQ.C) RETURN
23. C
24. C DATA SET LOOP
25. C
26. DO 150 I = 1,INDUSED
27. 150 IF (INPAR.EQ.C) GO TO 90
28. C
29. C CONVERT THE PARAMETERS
30. C
31. DO 80 L = 1,NPAR
32. J = IPAR(1,L) + 4
33. DO 80 K = 5,J
34. 80 CALL CHECK (PAR(K,L),IPAR(2,L),UNITS)
35. C
36. C COMPUTE BEGINNING SUBSCRIPT FOR DATA IN DATA SET NRS(I)
37. C
38. 90 RKT = 0
39. IF (NRS(1).EQ.1) GO TO 120
40. IK = NRS(1-1)
41. DO 110 IIA = 1,IK
42. 110 RKT = RKT + NROWS(IIA)
43. 120 II = NRS(1)
44. L7 = NROWS(11)
45. DO 140 L5 = 1,L7
46. 140 RKT = RKT + 1
47. C
48. C CONVERT THE VARIABLES
49. C
50. DO 130 L4 = 1,NVAR
51. DSUB = RKT + NVAR - NVAR
52. 130 CALL CHECK (DATA(DSUB+L4),IVAR(1,L4),UNITS)
53. 140 CONTINUE
54. 150 CONTINUE

```

RETURN
END

55.
56.

```

VELT,L      J,UPKREA,,UPANEA

1.      SUBROUTINE UPKREA(PORA,ILAT,ILON,IDAT,ITIM,IZON,
2.      ,FLAT,FLON,FDAT,FTIM,FZON,IOT)
3.      C
4.      C MAKE ALL VARIABLES INTEGER AND DIMENSION IOT.
5.      C
6.      C IMPLICIT INTEGER (A-Z)
7.      C DIMENSION IOT(4)
8.      C
9.      C UNPACK POINT OR AREA BIT.
10.     C
11.     C PORA=PP
12.     C IF (IOT(1),LT,J) PORA=A*
13.     C
14.     C LOOP TO UNPACK INITIAL DATA FIRST, THEN FINAL DATA.
15.     C
16.     C DO 2 I=1,3,2
17.     C
18.     C INITIALIZE INTERMEDIATE VARIABLES TO 0.
19.     C
20.     C IT1=J
21.     C IT2=0
22.     C IT3=0
23.     C IT4=0
24.     C IT5=0
25.     C SIGN=0
26.     C ITMP=0
27.     C
28.     C UNPACK THE DATE.
29.     C
30.     C ITMP=FLD(I,17,IOT(1))
31.     C IT1=ITMP/100
32.     C IT1=(ITMP-IT1*100)*100*IT1
33.     C
34.     C UNPACK THE TIME.
35.     C
36.     C IT2=FLD(I,12,IOT(1))
37.     C
38.     C UNPACK THE ZONE.
39.     C
40.     C FLD(I,5,IT3)=FLD(30,5,IOT(1))
41.     C
42.     C UNPACK THE LATITUDE.
43.     C
44.     C IT4=FLD(I,17,IOT(1+1))
45.     C
46.     C GET LATITUDE SIGN BIT, ADJUST LATITUDE ACCORDINGLY.
47.     C
48.     C SIGN=FLD(35,1,IOT(1))
49.     C IF (SIGN.GT.0) IT4=-IT4
50.     C
51.     C UNPACK THE LONGITUDE.
52.     C
53.     C IT5=FLD(I,18,IOT(1+1))
54.     C

```


55. C GET LONGITUDE SIGN BIT, ADJUST LONGITUDE ACCORDINGLY.

56. C
57. C SIGN=FLD(17,1,10111+11)
58. IF(SIGN.31.0) ITS=-ITS

59. C
60. C IF 1 IS NOT 1, ARE ARE ALMOST DONE.

61. C
62. C IF(1.EQ.1) GOTO 1

63. C
64. C FZAT=111

65. C FZIM=112

66. C FZON=113

67. C FLAT=114

68. C FLOR=115

69. C RETURN

70. C

71. C 1 IDAT=111

72. C I1IM=112

73. C IZON=113

74. C I1AT=114

75. C I1OR=115

76. C
77. C UNPACK THE FINAL POINT DATA.

78. C

79. C 2 CONTINUE

80. C END

BEST AVAILABLE COPY

BEST AVAILABLE COPY

```

1.  SUBROUTINE VOLUME(VALUE,UNICODE,UNITS)
2.  C
3.  C TABLE OF VOLUME MEASURE CONVERSION FACTORS.
4.  C
5.  C INTEGER ENTRY(3,4),UNITS(4),UNICODE
6.  C FACTORS ARE DOUBLE PRECISION.
7.  C
8.  C DOUBLE PRECISION FACTOR(4)
9.  C PUT UNIT CODES INTO ENTRY.
10. C
11. C DATA IENTRY(1,1),I(1,4)/17,10,19,20/
12. C
13. C PUT IN ALPHA UNITS.
14. C
15. C DATA IENTRY(1,3),I(2,3),J(1,4)/CM,0.3',0
16. C
17. C 'METERS',0.03 'FEET',0.03 '
18. C 'YARDS',0.03 '
19. C
20. C ENTER INTERNAL UNITS.
21. C
22. C UNITS(3)=METERS
23. C UNITS(4)=CU.
24. C
25. C
26. C IF UNICDE=1, THEN SET CODE TO STANDARD UNITS.
27. C
28. C IF(UNICDE.EQ.0) UNICDE=-18
29. C
30. C SET CONVERSION FACTORS.
31. C
32. C DATA IFACTOR(1),I(1,4)/1,00-6,10-2,83170-2,764500/
33. C
34. C IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
35. C CODE INTO 'UNITS'.
36. C
37. C ICODE=IABSTUNICODE)
38. C DO 2 I=1,4
39. C IF(ICODE.NE.ENTRY(I,1)) GO TO 2
40. C IF(UNICDE.EQ.0) VALUE=VALUE/FACTOR(I)
41. C IF(UNICDE.LT.0) VALUE=VALUE*FACTOR(I)
42. C DO 1 J=1,2
43. C UNITS(J)=ENTRY(J,1)
44. C 1 CONTINUE
45. C
46. C AT THIS POINT, CONVERSION IS COMPLETE.
47. C
48. C RETURN
49. C
50. C ELSE, CHECK REST OF TABLE.
51. C
52. C 2 CONTINUE
53. C
54. C IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.

```

52.
53.
54.
55.
56.
57.
58.
59.
60.
61.

CALLS(11) :
CALLS(21) :
PRINT 12, ICODE
FOR FURTHER E (CON--UNIT CODE, 13, NOT IN VOLUME TABLE.)
RETURN
END

BEST AVAILABLE COPY

DEL.L

N,NEST,NEST

```
1.      C
2.      C
3.      C
4.      C
5.      C
6.      IF (ABS(A-B) .GT. 183066) GO TO 10
7.      NEST = MIN(A,B)
8.      RETURN
9.      C
10.     NEST = B
11.     IF (A .GT. B) NEST = A
12.     RETURN
13.     END
```


APPENDIX B
CURRENT NAVDAB STEERING GROUP

APPENDIX B

CURRENT NAVDAB STEERING GROUP

The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is under the general sponsorship and direction of NAVSEA 06H1-4. NAVDAB functions as a subsidiary of the Mobile Sonar Technology (MOST) Acoustics of the Medium Committee (ACME) for NAVSEA 06H1. Dissemination of information and requests for any of the documents pertaining to NAVDAB may be obtained by sending a formal request to any of the following current NAVDAB Steering Group members. A list of official mailing addresses follows the list of Steering Group members.

Executive Secretary

A. P. Franceschetti, SEA 06H1-4
Naval Sea Systems Command
Washington, D.C. 20362
Commercial (202) 692-3166
Autovon 222-3166

Chairman

L. A. Smothers, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Branch Managers

F. R. DiNapoli, Code TA113
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2647
Autovon 636-2647

Branch Managers (Continued)

W. H. Geddes, Code 3440
Naval Oceanographic Office
Washington, D.C. 20373
Commercial (202) 433-3994
Autovon 288-3994

L. A. Smothers, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Steering Group

F. R. DiNapoli, Code TA113
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2647
Autovon 636-2647

R. H. Ferris, Code 8120
Naval Research Laboratory
Washington, D.C. 20375
Commercial (202) 767-3359
Autovon 297-3359

W. H. Geddes, Code 3440
Naval Oceanographic Office
Washington, D.C. 20373
Commercial (202) 433-3994
Autovon 288-3994

R. F. Hosmer, Code 3071
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Steering Group (Continued)

W. A. Kuperman, Code 8120
Naval Research Laboratory
Washington, D.C. 20375
Commercial (202) 767-3210
Autovon 297-3210

K. V. Mackenzie, Code 9420
Naval Oceanographic Office
Naval Research Laboratory
4555 Overlook Ave. SW
Washington, D.C. 20375
Commercial (202) 767-2830
Autovon 297-2830

L. C. Maples, Code TA1A
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2501
Autovon 636-2501

S. R. Santaniello, Code TA111
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2675
Autovon 636-2675

J. A. Whitney, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

OFFICIAL MAILING ADDRESSES

1. Commander
Naval Undersea Center
San Diego, California 92132
2. Commanding Officer
Naval Underwater Systems Center
New London, Connecticut 06320
3. Director
Naval Research Laboratory
Washington, District of Columbia 20375
4. Commander
Naval Oceanographic Office
Washington, District of Columbia 20373
5. Commander
Naval Sea Systems Command
Washington, District of Columbia 20362

USER COMMENTS FORM

Comments concerning this document are invited. Please send one of the following attached forms or a letter to:

L. A. Smothers, Code 307
Naval Undersea Center
San Diego, California 92132

(Envelope address: Commander, Naval Undersea Center, San Diego,
California 92132)

MEMORANDUM

Date: _____

From: _____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST- 5 (Volume 4: Details of Retrieval Phase)

CUT

MEMORANDUM

Date: _____

From: _____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST- 5 (Volume 4: Details of Retrieval Phase)

CUT

MEMORANDUM

Date: _____

From: _____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST- 5 (Volume 4: Details of Retrieval Phase)

CUT

DISTRIBUTION LIST

- 5 Naval Sea Systems Command
 NSEA-06H1 (5 cys)
- 3 Naval Oceanographic Office
 Code 3432
 Code 3440
 Code 6100
- 2 Naval Research Laboratory
 Code 8120 (2 cys)
- 7 Naval Undersea Center, San Diego
 Code 307 (7 cys)
- 6 Naval Underwater Systems Center, New London Laboratory
 Code TA1A
 Code TA42
 Code TA111 (2 cys)
 Code TA112
 Code TA113
- 2 Defense Documentation Center

FILM
5